The Architecture and Design of a

SONET Receive-side Overhead Processor for OC-48

by

Ira Claude Denton

Submitted to the

Department of Electrical Engineering and Computer Science

in partial fulfillment of the requirements

for the degrees of

Bachelor Of Science

and

Master Of Science

at the

Massachusetts Institute Of Technology

June, 1994

Signature of Author _____
Department of Electrical Engineering and Computer Science
April 22, 1994

Certified by _____
Dr. Vincent Chan
Thesis Supervisor (Academic)

Certified by _____
Dr. Jim Gimlett
Company Supervisor (Tektronix)

Accepted by _____
F.R. Morgenthaler
Eng. Chair, Department Committee on Graduate Students

The Architecture and Design of a

SONET Receive-side Overhead Processor for OC-48

by

I. Claude Denton

Submitted to the

Department of Electrical Engineering and Computer Science

April 22, 1994

in partial fulfillment of the requirements for the degrees of

Bachelor Of Science and Master Of Science in Electrical Science and Engineering

ABSTRACT

A 2.488 gigabit/second digital network adhering to the SDH (Synchronous Digital Hierarchy) or SONET (Synchronous Optical Network) standard creates and terminates tens of megabits of overhead per second. The receive side of each node in the network must be able to capture, interpret, and react to various defined channels within this overhead stream. At many nodes it may also be necessary to reorganize the data stream in order to perform add/drop or multiplexing functions. This thesis describes the architecture and design of a single-chip section and line terminating element that provides receive-side overhead processing and STM-1/STS-3-level reorganization functions.

Thesis Supervisor (MIT): Dr. Vincent W. S. Chan

Title: Associate Head, Communications Division, MIT Lincoln Laboratory

Thesis Supervisor (Tektronix): Dr. Jim Gimlett

Title: Program Manager, Advanced Communications, ERL, Tektronix, Inc.

Acknowledgments

My first thanks go to the people of the Tektronix Electronics Research Lab who made this thesis project possible, and supported and guided me throughout my research. Jim Gimlett and Bruce Murdock created the OC-48 project and my place in it, and ignited my enthusiasm for the project with their own. Jim also provided me with invaluable insight into the SONET standard and technical assistance in developing the architecture of a processing element to fit it. Vallath Nandakumar acted as a sounding board and troubleshooter throughout the design phase of my project, and has taken over its layout. I am also grateful to the rest of the group for making me feel at home in ERL; I'm looking forward to returning on a permanent basis this year.

I also want to thank Dave McKinney of the Tektronix Advanced Design Group for helping me learn to use his group's powerful design tools and answering my questions at every turn. Finally, thanks to Ray Veith for providing me with the extra motivation to write my thesis in a timely manner.

On the other coast, I'd like to express my gratitude to Vincent Chan for the support and guidance he has provided as my MIT thesis supervisor, and to Mitchell Trott for helping me understand the probability of finding a pattern in a random data stream.

Finally, I'd like to thank my parents and grandfather Denton for making it possible for me to come to MIT and for supporting me emotionally, and sometimes technically, throughout my time here. Special thanks to my friends and fiancé for making it fun.

Table Of Contents

List Of Figures

I. Introduction

The Synchronous Optical Network (SONET) and Synchronous Digital Hierarchy (SDH) standards [1,2,3] specify rates and formats for synchronous digital optical communication networks intended for use in telephony and at the lowest layer of Integrated Services Digital Networks (ISDNs). They establish a hierarchy of data transfer rates extending from a level that might be required by a small group of users (51.84 Mb/s), to a level that is suitable for long-haul carrier operations (2.488 Gb/s). The expected uses of the networks demand provisions for multiplexing lower-rate signals together into a higher rate signal for long-distance transmission, monitoring system performance to ensure reliability, and communicating maintenance information among nodes in the network without interrupting normal service. The synchronous nature of the network necessitates a means of extracting timing information from an incoming data stream and recognizing timing failures. The standards answer these concerns by establishing a frame structure within the transmitted signal and dividing each frame into a payload and an overhead section. The frame structure supports multiplexing by allowing high rate signals to be specified in terms of combinations of lower rate signals. The payload section is filled with the data to be transmitted across the network, while the overhead section allows the insertion of synchronization strings, parity check bytes, and maintenance communication channels [4].

Different elements in the transmission network need access to different amounts of knowledge about the data being transmitted. An end receiver, for instance, must establish synchronization with the frame structure of the incoming signal, check it for data integrity, process any maintenance messages that may accompany it, pass information to its companion transmitter, and extract the payload data. A mid-network regenerator need only synchronize with the data, check for and signal error conditions, and re-transmit it. A multiplexer lies somewhere in between. The standards codify these differences by

establishing four layers of access to the SONET/SDH stream: photonic (or physical), section, line, and path. An element that changes the stream at one of these levels is called a terminating element of that layer, and is required to interpret the associated overhead. Figure 1 depicts the SONET 2.488 Gb/s (OC-48) frame structure and the named section and line overhead bytes.

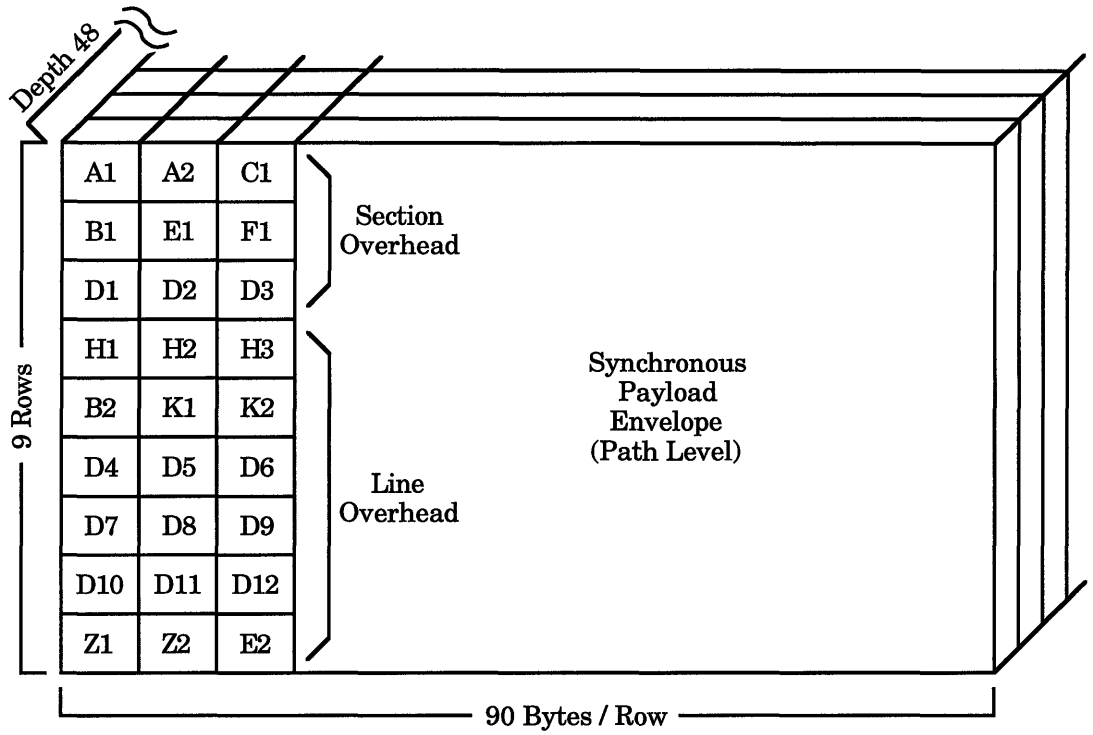


Figure 1: 0C-48 Frame Structure and Overhead

Conceptually, the frame of any higher-rate signal in the SONET hierarchy is built by byte-interleaving frames of 51.84 Mb/s (OC-1) signals. Thus an OC-48 frame can be depicted as a three-dimensional structure of depth 48, where each tier is an OC-1 frame, as in Figure 1. An OC-1 frame is transmitted one byte at a time, stepping along the rows (i.e. A1, A2, C1, ...,B1, E1, F1, ...), so the byte interleaving process used to construct higher-rate frames results in a transmission indexed first by depth, then along the rows (i.e. A1, A1, ..., A2, A2, ...). This formalization allows the time duration of a SONET frame to be the same at every rate, 125 μs, and ensures that the bytes of the framing pattern (A1, A2) are always

transmitted at the beginning of the frame. While it is not within the scope of this thesis to discuss the function of the section, line, and path overhead in detail, a brief statement of the functions of the named section and line overhead bytes for OC-48 is relevant to the topic.

Many of the overhead bytes are defined only for the first OC-1 tier of a higher rate signal; for instance, the byte behind the B1 byte in Figure 1 serves no defined function. Others are defined for all tiers. In the following discussion, the former case is to be assumed unless the latter is explicitly noted [4,5]. The section overhead consists of the named bytes A1, A2, C1, B1, E1, F1, and D1-D3. The A1 and A2 bytes together form the pattern by which the beginning of a frame can be recognized, and as such are defined for all 48 OC-1s of an OC-48 signal. They always hold the same values: A1 = F6, A2 = 28 in hexadecimal notation. The C1 byte was originally designated as an OC-1 identification byte, defined for all OC-1s, but is currently under study for use as a section trace indicator to identify physical connections, defined only for the first 16 tiers. The B1 byte is used to perform a single eight-bit bit-interleaved-parity error check covering the entirety of the previous OC-48 frame. The E1 byte carries the section orderwire channel, defined as a 64 kb/s serial channel (8 bits per frame, 8000 frames per second) for network equipment communications. The F1 byte is designated for use as a 64 kb/s user channel for implementation-specific purposes. The D1-D3 bytes form the section data communication channel, a 192 kb/s control channel for section layer network elements.

The line overhead consists of the H1-H3, B2, K1-K2, D4-D12, Z1-Z2, and E2 bytes. Bytes H1, H2, and H3 are defined for all 48 tiers of the frame and are used as a pointer to and a stuff byte for the payload data. The B2 byte is also defined for all 48 tiers; each of these 48 bytes serves as an eight-bit bit-interleaved-parity checksum for the line overhead and payload envelope of the STS-1 in the same tier of the previous frame. The K1 and K2 bytes carry the automatic protection switching channel. The line data communication channel is carried in bytes D4-D12. It is a 576 kb/s channel analogous to the section data

communication channel. The Z1 and Z2 bytes are designated as growth bytes in all 48 tiers. Only two have thus far been explicitly defined: the first Z1 byte is to be used as a synchronization status byte, redesignated S1, and the third Z2 byte, redesignated M1, is to be used to transmit a far-end-bit-error count back from remote receivers on two-way links. The E2 byte carries the line orderwire channel, a 64 kb/s channel analogous to the section orderwire.

A significant number of useful intra-network entities perform processing at the photonic, section, and line levels, but do not interact with payload data, i.e. the path level. These include all varieties of multiplexers. In addition, all elements that do manipulate payload data must first resolve section and line overhead. A processor that can provide required overhead interpretation functions at the section and line levels, as well as some signal rearrangement at those levels, is thus a general and useful element. There are a few commercially available processors that perform section and line overhead termination at SONET rates up to 622 Mb/s (OC-12), and at least one that performs limited termination functions for OC-48. To the author's knowledge no available overhead processor offers signal rearrangement functions, or capture functions flexible enough to access new overhead bytes as the standards define them. This thesis describes an architecture that realizes such a processor for the SONET 2.488 Gb/s rate (OC-48), and details the design of the required functional blocks in high-speed CMOS standard cells.

In the interest of brevity, this paper will use only SONET terminology in most descriptions. Unless specifically noted, the discussion also applies to the corresponding SDH requirements. The abbreviations STS-N and OC-N are used to refer to the (possibly parallelized) SONET-derived stream with a total data rate of N times the base rate of 51.840 Mb/s. Thus OC-48 and STS-48 refer to a 2.488 Gb/s stream, OC-12 and STS-12 to a 622.080 Mb/s stream, and so on. This notation implies SONET framing conventions.

II. Architecture

1. System

In order to propose an architecture for a application-specific integrated circuit (ASIC), one must regard it in the context of a larger system. The requirements of that system as a whole and the capabilities of its other parts combine to specify the functions the ASIC must implement and the interface it must present. The host system of the processor ASIC described here is a general receiver for use in physical, section, and line terminating nodes in a SONET network. The SONET standard describes the receive functions required of such nodes. A physical terminator must provide optical-to-electrical and electrical-to-digital signal conversions according to strict specifications that ensure low error rates [6]. A section terminating element must perform framing, descrambling, overhead extraction, and error monitoring functions on the digital signal [7]. A line terminating element then provides multiplexing, protection switching, overhead recovery, and further error monitoring functions [7]. After all these operations have been performed, the payload of the signal may be extracted by a path terminating element, or the signal may be reorganized and transmitted further. Since the use of overhead processing operations varies among network elements, a general control and data interface between the receiver and the element in which it resides should be specified.

Given these required attributes, a number of system architectures could be proposed. The ASIC design presented here assumes a system partitioned according to clock rate. The 2.488 Gb/s data rate of the incoming OC-48 implies that physical layer termination will result in a 2.488 GHz clock synchronized to a serial data stream. The section terminating subsystem must contain extremely fast circuit elements to handle these signals. On the other hand, the complexity of logic and amount of memory required to implement section and line terminating functions make ease of design and circuit density

important considerations. These conflicting requirements can be satisfied by developing circuits in a high-speed, low-density device technology such as GaAs or high-speed bipolar Si that accept the serial stream and slow its clock rate through parallelization. The bulk of the processing functions can then be realized in a relatively low-speed, high-density technology such as CMOS. The availability of standard cell libraries and hardware description language development environments for CMOS ASIC design makes the implementation of these low speed circuits relatively easy. A diagram of the blocks resulting from this partitioning is shown in Figure 2.

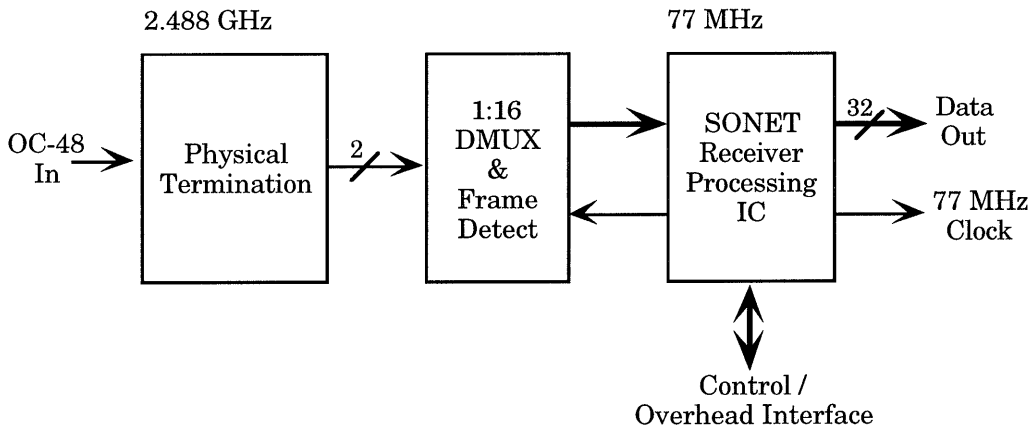


Figure 2: Block Diagram of Receiver System Architecture

The SONET Receiver Processing IC is the focus of this thesis. It is called the SONET Overhead Recovery and Cross Connect IC (SORCC).

2. The SORCC

To meet the requirements of the system, the SORCC supports three types of operations: data processing, overhead processing, and control. Its architecture recognizes these three categories and reflects them as three subsystems within the IC. Each subsystem function corresponds to a termination operation required by the SONET standard or an interface element required by the system architecture. This section will

present and justify the description and organization of the blocks and subsystems within the SORCC. Figure 3 illustrates the position of these subsystems with respect to the SONET stream.

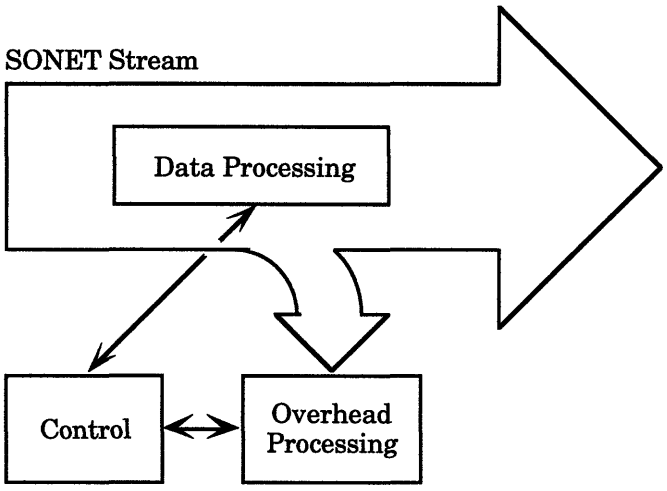


Figure 3: SORCC Subsystems and the SONET Stream

A. Data Processing

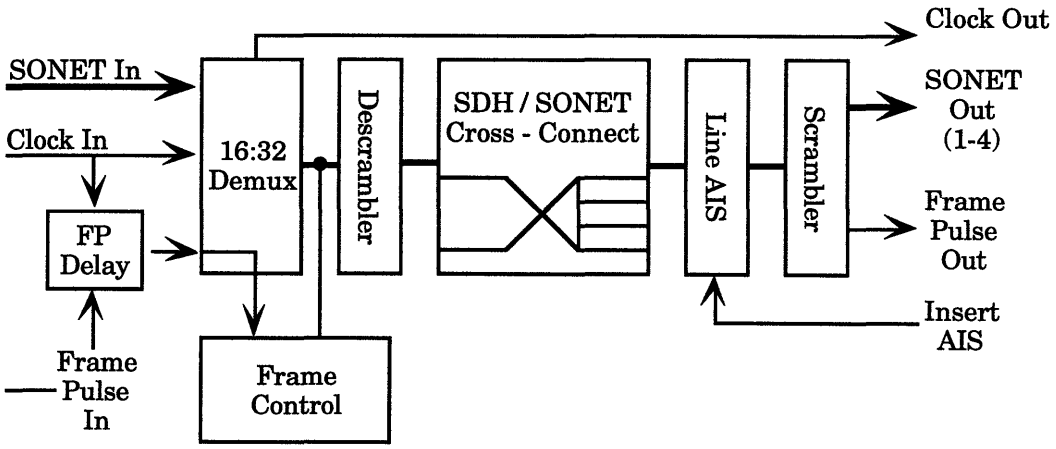


Figure 4: Data Path Functional Blocks

Figure 4 shows the blocks that make up the data path through the SORCC. The data path has as its inputs the 16-bit-wide version of the OC-48 stream, the 77 MHz dual-edged clock, and the frame pulse outputs of the 16:32 DMUX & Frame Detect block. The

frame pulse signal is of particular importance because most of the functional units of the SORCC depend on some version of it for synchronization to the SONET frame structure. The standard specifies two sets of 48 bytes at the beginning of each frame, the A1 and A2 bytes, to serve as a framing pattern for all network elements to recognize. Any scheme for synchronization utilizing some subset of this 96-byte pattern for frame detection and meeting certain timing and probability requirements is acceptable [8]. The SORCC is designed to work with a frame detect system that recognizes three bytes around the A1A2 boundary (byte 48 is the last A1, byte 49 the first A2). Most frame detect circuits available on the open market use such a strategy. A single check of a three-byte sequence does not meet the assurance level against false frame recognition required by the standards, as will be developed later, so a mechanism of screening out invalid frame pulses is provided in the SORCC.

The frame pulse (FP) delay block inserts a user-selectable delay of up to two clock periods, or cycles, in front of the incoming frame pulse, providing flexibility to the designer of the frame detect block. Because of it, the frame pulse can arrive in any of four positions relative to the start of the A2 overhead bytes and still be properly aligned for the SORCC to recognize. The 16:32 Demultiplexer (Demux) reduces the data rate down from 77 Mb/s per line, referenced to both edges of the 77 MHz clock, to 39 Mb/s per line, referenced to only the falling edge of the 77 MHz clock. This makes it possible to realize the more complex logic of the rest of the chip in a design using available standard cells. The frame control block performs one function that affects the data path: it creates a new frame pulse aligned to the first word of each frame. This frame pulse is used by all following blocks in the SORCC, so any spurious frame pulses created by the frame detect circuit are screened out before they cause problems in the SORCC.

The descrambler block is a 32-bit-wide implementation of the shift-register frame synchronous scrambler specified in the standards [9]. SONET frames are scrambled with a

pseudo-random sequence before transmission to ensure that, for virtually any payload, the data stream broadcast contains enough transitions to permit clock recovery at the receiver. This scrambling covers all of the data in a frame except for the first row of overhead, that is, the A1, A2, and C1 bytes. The descrambler performs the inverse operation on the same set. The descrambler can be deactivated, allowing data to pass through unmodified. The descrambler would probably only be turned off in a network element in order to perform system test functions.

The cross-connect is a user-configurable block that allows reorganization of the OC-48 signal at the STS-3 level [10]. It defines sixteen output and sixteen input "slots," each a byte wide, and provides a logical crossbar connection among them. The sixteen input slots are filled by sets of four 32-bit words from the input STS-48 data stream. By aligning the first of these four words to the first word of the frame, the cross-connect assures that the 16 bytes contained by the four consecutive words are in order from STS-3 #0 to STS-3 #15, so the input slots can be numbered according to the STS-3 that passes through them. Each of the 16 output slots also occupies a fixed position within the output stream. The data coming in on any of the input slots can be sent out through any of the output slots, allowing any STS-3 from the incoming STS-48 to be put out through any (or any combination) of the STS-3 slots in the outgoing 2.488 Gb/s stream. Providing this function in general receivers and transmitters greatly simplifies the task of building of add-drop and general multiplexing network elements, since any desired channel from the SONET stream can be output in any position from the receiver and accepted in any position at the transmitter. For reference, the structure of an STS-48 signal in terms of lower-rate signals is shown in Figure 5. A graphical depiction of the "slots" described above is shown in Figure 6.

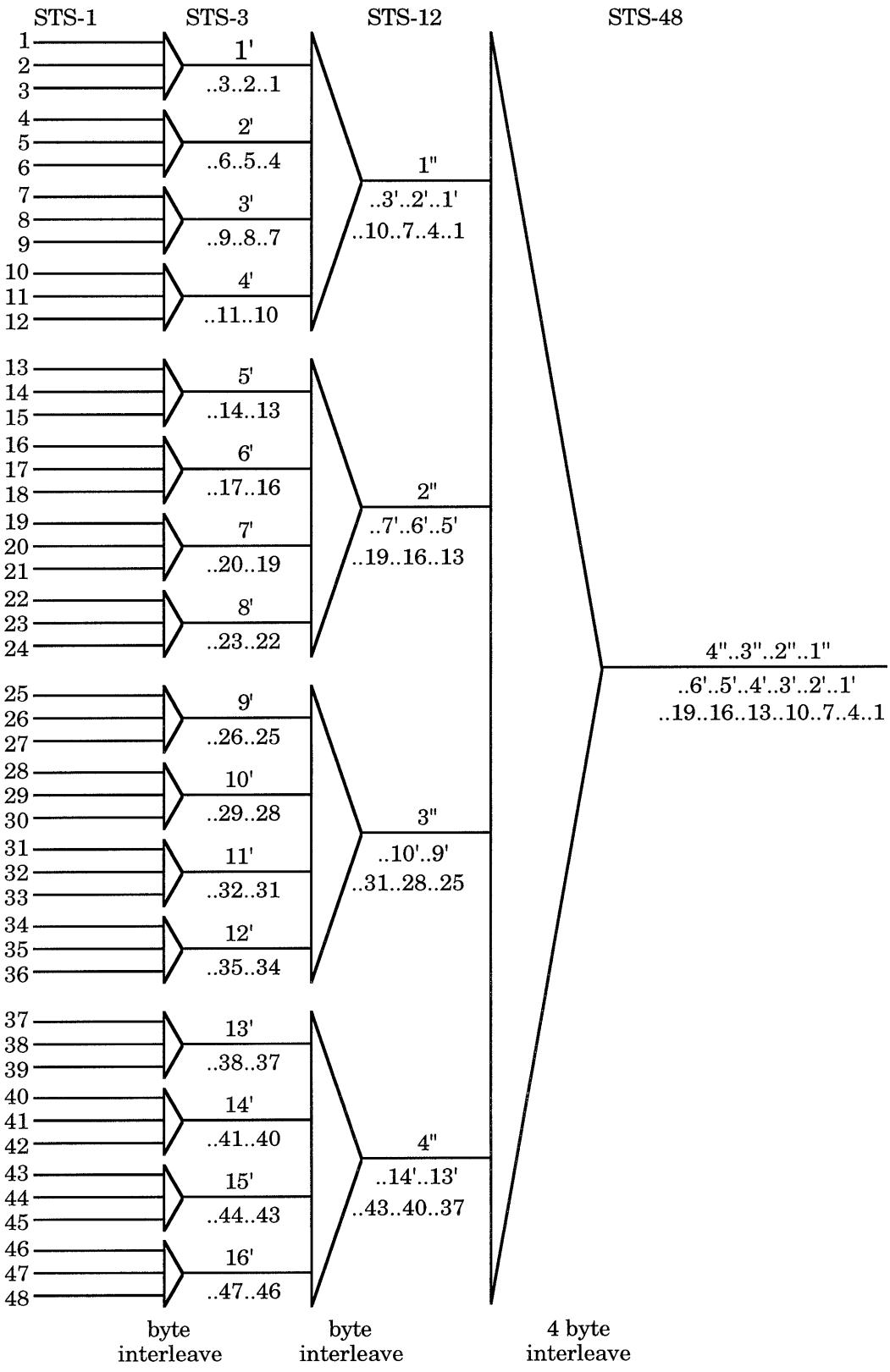


Figure 5: Creation of an STS-48 Signal from STS-1 Signals

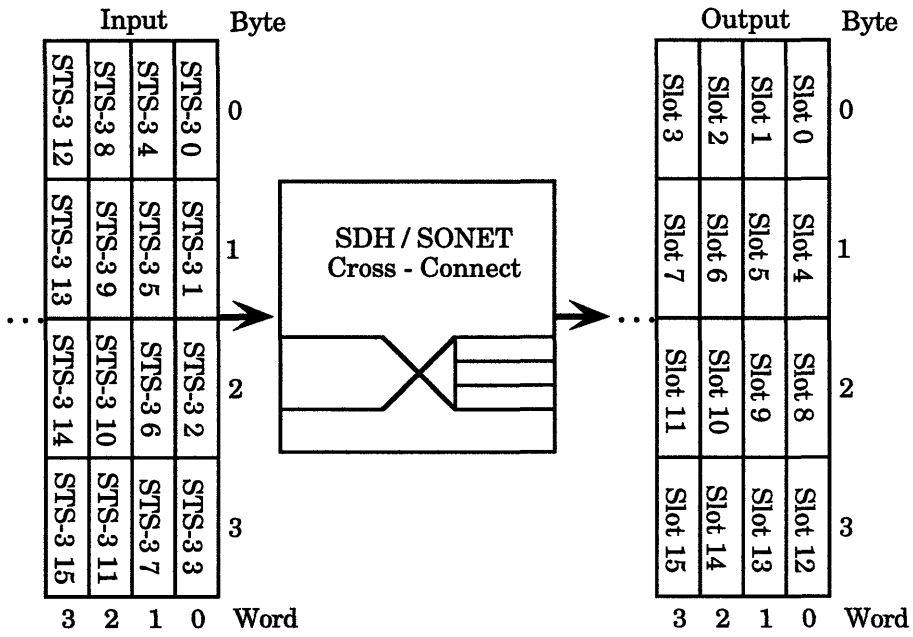


Figure 6: Cross Connect Input and Output Slots

The Line AIS block inserts the alarm indication signal (AIS) [11] into the outgoing SONET data stream. This signal is comprised of an all-ones pattern filling the entire frame except for the section overhead. It may be inserted as a response to problems such as loss-of-frame (LOF) and loss-of-signal (LOS) to communicate to a path-terminating element that a failure has occurred. An external pin is used to activate this block so as to allow the host system to choose whether or not to overwrite the existing data stream in these error cases.

The scrambler block is a set of four byte-wide frame-synchronous scramblers based on the same SONET scrambling polynomial as the descrambler. If this block is enabled, it forces the chip's four output channels into the STS-12 format, though the SORCC does not perform some of the functions that guarantee recovery of legal STS-12s from an STS-48. The scrambler is present in this IC in order to create signals loosely suitable for direct serialization and retransmission from an OC-48 to OC-12 demultiplexer. In a general network element, the signal coming out of the SORCC would probably be processed further before being retransmitted, so the scrambler would be disabled.

To summarize: on its trip through the data path of the SORCC, the STS-48 signal is spread from a 16 bit wide 77 MHz signal to a 32 bit wide 39 MHz signal, realigned to the frame pulse, optionally descrambled, reorganized at the STS-3 level, possibly overwritten by an alarm indication signal, and optionally scrambled for output as four channels of STS-12. All of this processing must maintain a throughput of 2.488 Gb/s and as low a latency as possible.

B. Overhead Processing

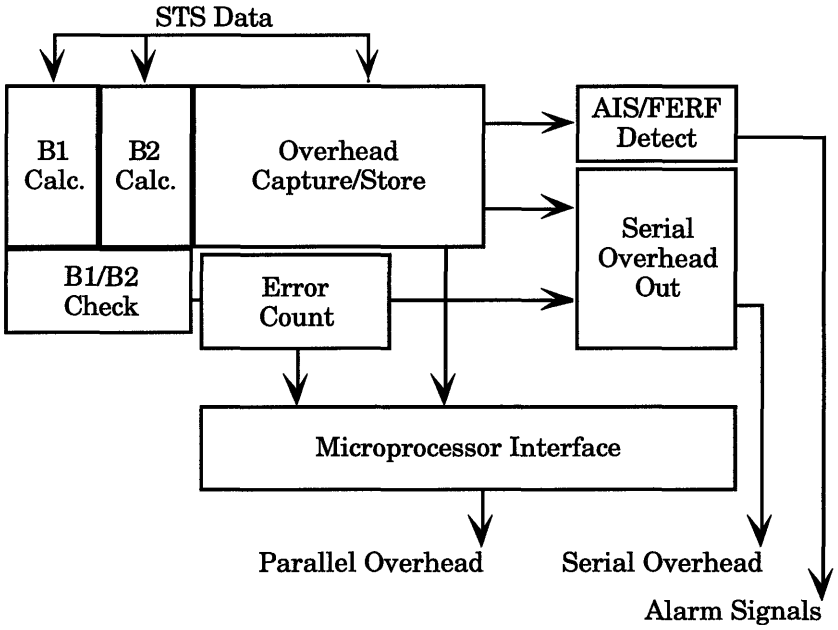


Figure 7: Overhead Processing Blocks

Figure 7 depicts the overhead processing subsystem of the SORCC. Overhead functions fall into three categories: overhead-based error monitoring, overhead capture, and overhead presentation. Overhead-based error monitoring entails computing the B1 and B2 parity counts defined in the SONET standard [12], comparing them to the values sent from the transmitting end, and keeping a count of mismatches. Devoted to these functions are the B1 and B2 calculation blocks, the B1/B2 check block, and the error count block.

The parity calculations required by the SONET standard yield a byte or set of bytes with bits such that a Boolean addition of any bit of the calculated value with all the bits in the same position within the covered data set yields a value of zero. As an example, the even parity byte corresponding to the bytes 01010101, 11001100, 00011000 would be 10000001. The B1 is a one byte parity calculation covering the entire frame's scrambled data. The B2 is a forty-eight byte parity calculation, where each byte covers one STS-1's data, excluding the section overhead, before that data has been scrambled. The B1 calculation block in the SORCC looks at the four-byte-wide data stream passing between the 16:32 demux and the descrambler, and computes a byte that yields even parity over all the bytes of one frame of that stream. While it is computing the byte for one frame, it makes its calculation for the previous frame available to the B1/B2 check block. The B2 calculation block looks at the data passing between the descrambler and the cross-connect and computes 48 bytes that yield even parity over all bytes not belonging to the section overhead of each frame. As in the case of the B1 calculation, the last frame's parity words are made available for checking as the current frame's parity is accumulated.

The B1/B2 check block compares the data sent in each frame's B1 and B2 overhead bytes with the parity bytes calculated by the B1 and B2 calculation blocks during the previous frame. It sources a count of the number of bits that do not match, and a line that indicates whether the count is of B1 or B2 errors. The count output of this block is forced to zero when the SORCC has not yet detected the frame structure, to avoid accumulating a false error count. The error count block accumulates the count sourced by the B1/B2 check block during each frame and updates its internal error counters on each frame pulse. There are three counts available from this block: accumulated B1 errors, accumulated B2 errors, and errored frames. The errored frame count is incremented by one for each frame in which one or more B1 or B2 errors are detected. All three counters can accumulate at least a second's worth of errors without overflow. The internal counters can be latched into a user-readable space and cleared through the control interface.

The overhead capture / store block performs all of the overhead capture functions of the SORCC. Every frame, it captures the following overhead bytes: the first 16 C1s, which are under study for use in a section trace function; the synchronization status, also under study [5]; the section and line data communication and orderwire channels; the section user channel; the automatic protection switching (APS) channel; and the far end bit error (FEBE) byte. These bytes are defined in [12]. Also, in recognition of the fact that many networks may use some of the undefined of the overhead bytes to establish application-specific channels, and that the standards themselves are still in flux, 24 user-selectable bytes are captured from each frame. These bytes can come from anywhere within the frame, overhead or payload. As well as capturing overhead data, the overhead capture block performs the filtering of the APS channel required by the standards [13]. It keeps separate APS values that are only updated when the same value is captured in the K1 or K2 byte for 3 or 5 consecutive frames, as selected by the system. The alarm indication signal / far end receive failure (AIS/FERF) detect block sources alarm signals based on this filtered value.

All of the error monitoring and capture functions exist to recover administrative information from the STS-48 without interfering with its further transmission. The serial overhead output, the parallel overhead output section of the microprocessor interface, and the AIS/FERF detect blocks serve to provide access to this information. The serial overhead output block provides dedicated output pins for the defined serial channels: the section and line orderwire and data communication channels and the user channel. It also outputs in serial form single-frame B1 and B2 error counts for use by the network element's transmitter side to build a FEBE count, and a serial stream of the 24 user-selectable capture bytes for application-specific uses. During each frame, these serial lines output the data stored or computed during the previous frame. The microprocessor interface block represents the general control and data interface described in section II.1. Although much of the SORCC's memory is distributed, this conceptual block codifies the presentation of a parallel interface for all access functions. The captured overhead bytes and the accumulated error counts are

available through it. The parallel interface also supports a "snapshot" option that allows the user to inhibit the normal frame-by-frame update of captured overhead, so the host system can take longer than one frame (125 μs) to survey the data, which could prove useful during system debugging and design. This snapshot affects both the serial and parallel presentation of captured data. Finally, the AIS/FERF detect block provides dedicated pins that indicate the presence of these alarm signals in the APS channel immediately, without processing by the external system. Together, these three interface blocks provide considerable flexibility to the network element designer.

C. Control

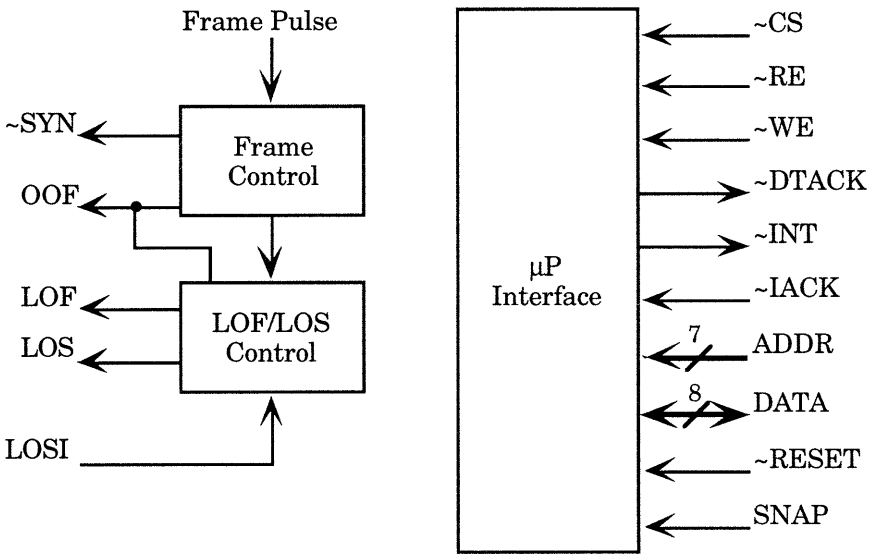


Figure 8: Control Blocks

The blocks that define the two-way SORCC-to-system interface are those of the control system. They serve to keep the SORCC synchronized to the rest of the system and support interactive communication with other devices. These are the frame control, LOF/LOS control, and microprocessor interface blocks, shown in Figure 8. The frame control and LOF/LOS control blocks serve to determine frame and signal related states as described in the standards, while the microprocessor interface works mostly to support features associated with exchanging information with the host network element.

The frame control block implements a state machine that uses input frame pulses and the system clock to determine frame alignment as defined in the standards [8]. As previously noted, it also sources internal frame pulses that are retimed to the first byte of each frame. It has two outputs: the OOF line, which indicates whether the system is operating in the out-of-frame (OOF) or in-frame (IF) state, and the synchronization line (~SYN), which is used to indicate to the 1:16 demultiplexer / frame detect system that synchronization has been lost and half-word realignment should be performed on the next detected framing pattern. The 16:32 demux block inside the SORCC performs word realignment based on this signal as well. These realignments are accomplished by recognizing that the first A2 byte should be the first byte of a half-word, and the fifth A2A2 half-word should be the first half of a full word. On the first A1A2 boundary detected after a falling edge of the ~SYN signal, the frame detect circuitry must realign its 16-bit output half-word so as to put the A2 in the high byte, and source a frame pulse synchronized to the new frame structure. Because of the action of the frame pulse delay block, that pulse always reaches the 16:32 demux block aligned with the fifth A2A2 half-word, so the 16:32 demux puts the half-word arriving along with the first detected frame pulse after a falling edge on the ~SYN signal at the top of a word. At all other times, word alignment must be held constant regardless of incoming framing patterns, since only the frame control block has enough information to judge those patterns valid or spurious.

The loss-of-frame / loss-of-signal (LOF/LOS) control block performs additional synchronization status monitoring as dictated by the standards. The loss-of-frame state will be entered if the SORCC remains in the out-of-frame state (OOF line high) for 3 ms without staying in-frame (OOF line low) for 3 ms continuously. That state will remain active until the IC stays in-frame for three continuous milliseconds. While in the loss-of-frame state, the LOF line will be asserted high. This implements the integrating timer described in the Bellcore SONET specification [14]. The loss-of-signal state will be entered

when the LOSI input line is asserted, indicating that upstream electronics have not seen an electrical transition in more than 2.3 μs. The SORCC will remain in that state until two frame pulses judged valid by the frame control block have been received [15]. The LOS output line indicates the presence or absence of the loss-of-signal state.

The microprocessor interface performs five key functions. First, it houses the master control register which contains "switches" that control the operation of several other blocks. Second, it monitors several status signals and provides the option of sourcing microprocessor interrupts based on them. Third, it implements the snapshot decision circuitry that controls the update of the overhead capture block. Fourth, it generates the data acknowledgment handshake signal associated with parallel register accesses in some microprocessors and microcontrollers. Finally, it bears responsibility for the specification of the parallel interface strategy that connects the visible register space of the SORCC to the host system.

Figure 9 shows how the blocks of these three major subsystems come together to form the complete SORCC processor.

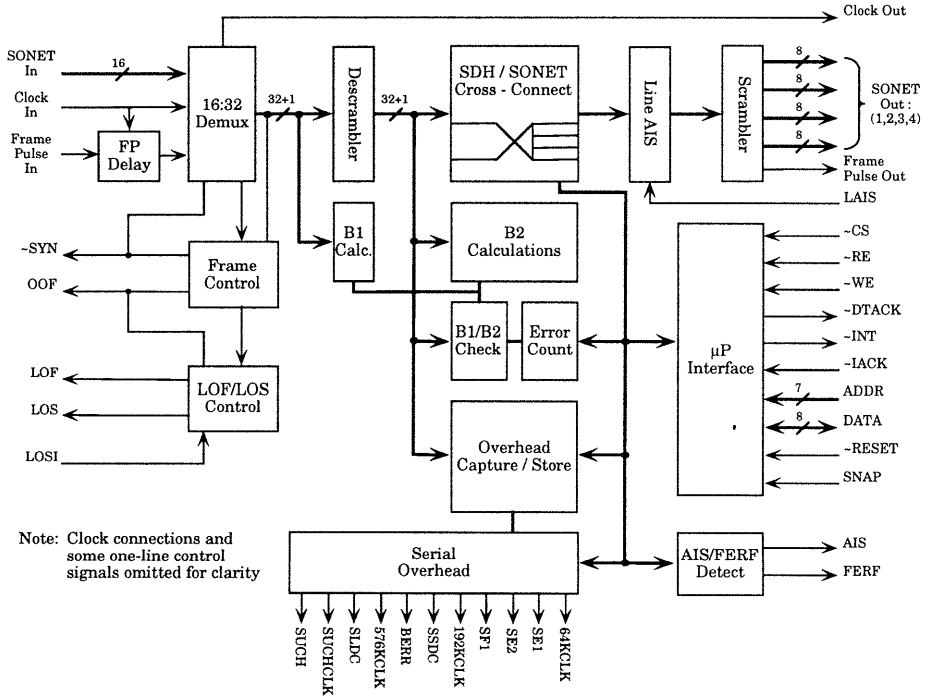


Figure 9: SORCC block diagram

D. Communication

An architecture should specify not only partitioning of a problem into subtasks, but also the means by which the units performing those subtasks may communicate. Communication between the SORCC and the host system is strictly specified by its input/output pins, listed in appendix 1, and its seven-bit address space, mapped in appendix 2. Communication among the functional units within the SORCC also needs to be clearly defined to maintain the level of abstraction represented by the block diagram of Figure 9.

The data path blocks can be viewed as a set of functions that operate in series on the data path. As such they communicate only through their input and output variables, the SONET stream and the frame pulse. No one block has access to the internal state of another.

The blocks of the overhead path, on the other hand, act to monitor and interpret the SONET stream, and thus need other communication paths, as well as a listening path to the stream itself. These connections are evident in Figure 7. The AIS/FERF block has direct access to the filtered K2 value inside the overhead capture and storage block to allow it to perform its interpretive role. The error count accumulators are connected directly to the six-bit error count sourced by the B1/B2 check block. There are also three more complex communication paths among blocks: the parity error check bus, the serial data interface bus, and the parallel system bus. The parity error check bus is a unidirectional communication link that transfers words of data from the parity calculation blocks to the B1/B2 check block. The check block is the dedicated master; it uses five address lines to specify which calculated parity value should be put on the 32 bit data bus. The responsibility for recognizing whether all or part of that data word is valid falls on the master. The serial data interface bus is also unidirectional; it carries bytes of data from the overhead storage block and the error count block to the serial overhead output block. Again the recipient of

the data is bus master, using a 6-bit address to call out the appropriate byte of data. The most complex link is the bidirectional parallel system bus, since it requires both a read and a write strategy. This bus provides access to control functions, configuration options, error counts, and captured data within the SORCC from the host system. The responsibility for codifying its access strategy belongs to the microprocessor interface, which sits symbolically between the host system and those internal blocks accessible in the SORCC address space. The interface strategy chosen is similar to that of standard RAM and ROM chips, and supports both popular microprocessor access protocols. The address spaces of all three SORCC busses are mapped in appendix 2.

The blocks of the control path require only point-to-point communication links. The microprocessor interface sends out control signals to several blocks, and the frame control block informs the LOS/LOF block whenever it detects a valid frame pulse or goes out of frame. All other control communications pass through the parallel system bus.

III. Design

Given the division of functions and definition of communication paths laid out by the IC architecture, one can perform a block-by-block design with good confidence that sub-units that work as the architecture specifies will combine together into a working realization of the SORCC. The modularity provided by defined communication paths inside the IC ensures that changes made inside one block will be invisible to others. The individual modules were designed using a Tektronix proprietary hardware description language that compiles to a netlist of standard cells in the Tektronix CMOS library. Where circuit diagrams are presented here, they were extracted from these netlists, not used to create them, so they show points of interest within the circuitry of a block rather than the entire block's schematic. All functional units of the SORCC were simulated individually using a Tektronix proprietary digital simulator with estimated parasitic capacitance values and shown to function as required. Whole-IC simulations have verified that they work together properly. The following sections will present the design of each block, working from top to bottom, left to right through the diagram of Figure 9.

1. Frame Pulse Delay

The frame pulse delay block decodes its setting from two dedicated control lines from the microprocessor interface. Based on that setting, it inserts the proper delay into the frame pulse channel to align its output pulse with the fifth 16-bit A2A2 word. To accomplish this, the block uses flip-flops to sample the frame pulse input (FPI) line on every clock edge and keep a history one full cycle deep. The decoded setting determines which of these four samples drives the output, synchronized combinatorially to the proper clock edge. Figure 10 depicts this function.

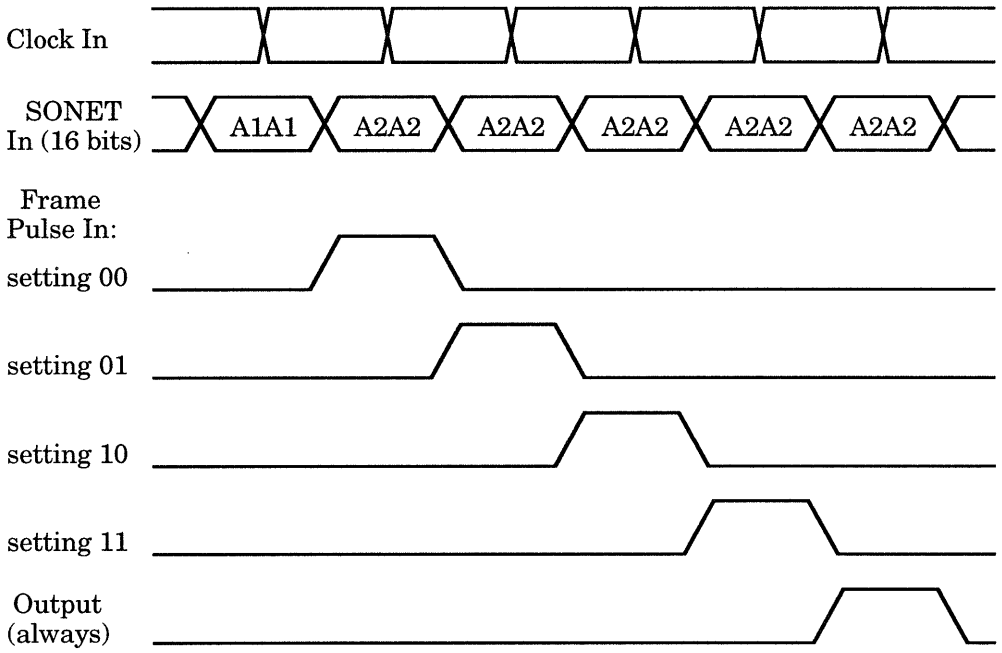


Figure 10: Frame Pulse Delay Function

2. 16:32 Demultiplexer

The 16:32 Demux block is composed of three 16-bit registers that provide three samples of the incoming data: on a rising edge, a once-delayed rising edge, and a falling edge. Likewise, three one-bit registers provide three samples of the incoming frame pulse line. The internally-generated swap signal controls one-of-two selectors that determine how the output signals will be configured. If the swap signal is high, the upper half-word of the 32-bit STS output is composed of the falling edge data sample, and the lower half-word is the rising edge sample. In this case, the output frame pulse line is driven by the falling edge frame pulse sample. If the swap signal is low, the delayed rising edge data sample drives the top bits of the data output, and the falling edge sample drives the low. Now the frame pulse output is generated from the delayed rising edge frame pulse sample. The swap signal is generated when the block detects the first frame pulse after the ~SYN pin has gone low (requesting byte realignment). After that, the swap signal is held constant until another falling edge of ~SYN occurs.

Thanks to the FP Delay block, the frame pulse is guaranteed to arrive aligned to the fifth A2A2 word. That 16 bit half-word should become the high half-word of the third A2A2A2A2 word inside the SORCC. Therefore, if the frame pulse is detected on a rising edge, swap should be set low to put a rising edge sample in the high bits of the data word, and subsequent frame pulses must also come on rising edges. Otherwise, swap should be set high to put the falling edge sample in the upper half-word and future frame pulses should come from falling edge samples. Since all other SORCC blocks sample data on only the falling edge of the 77 MHz clock, the delayed sample is required on the rising edge. Figure 11 shows the samplers and selectors.

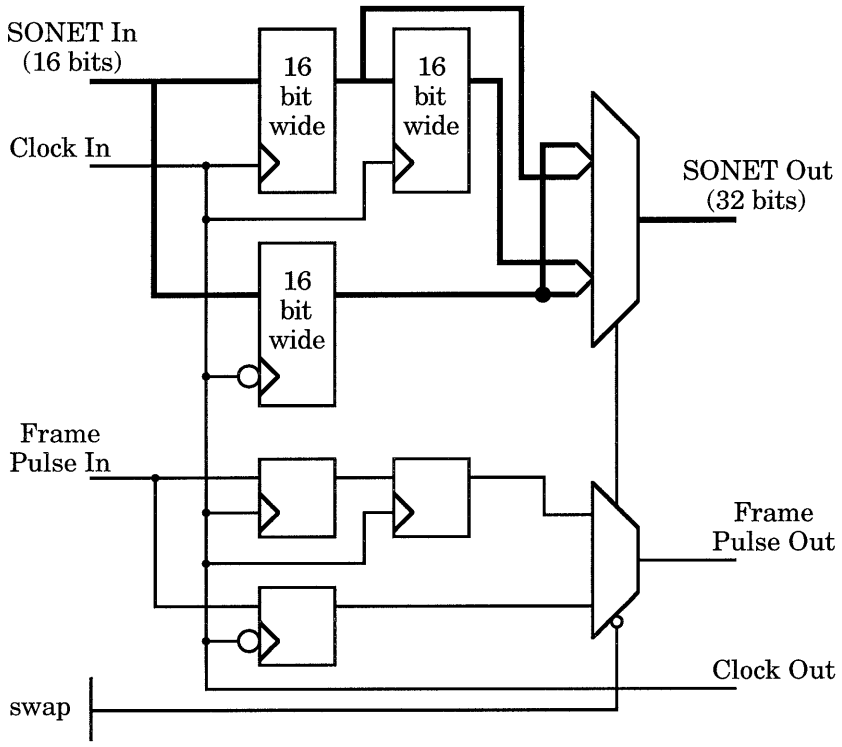
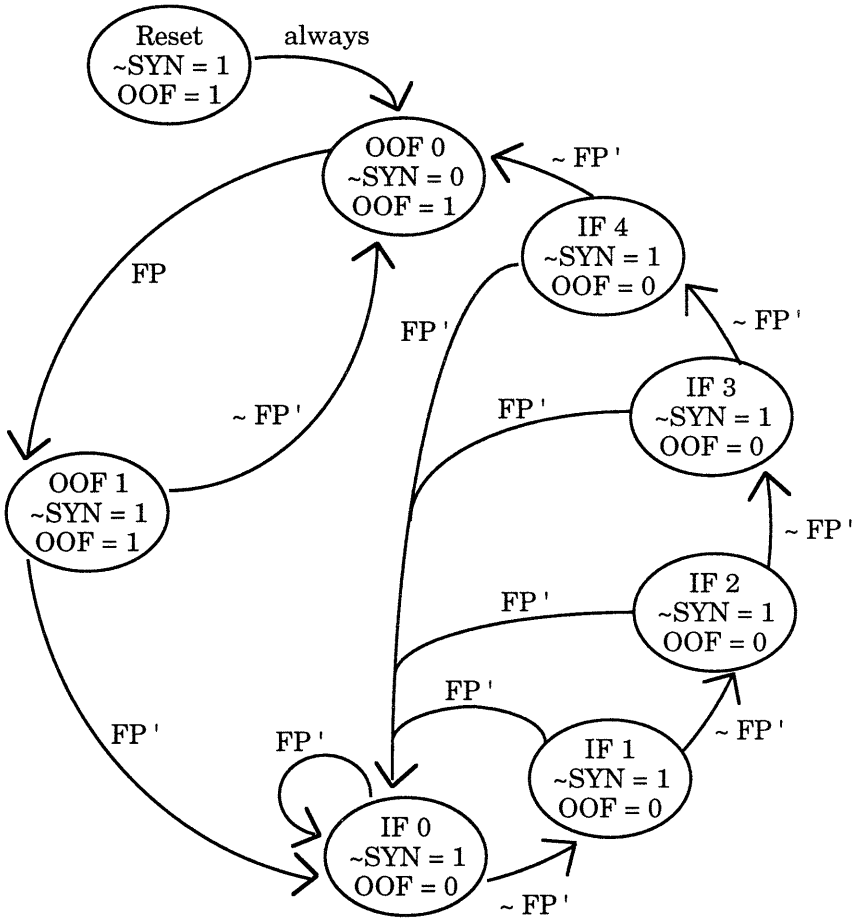


Figure 11: Samplers and Selectors of 16:32 Demultiplexer

3. Frame Control

The frame control block implements a state machine that determines in-frame and out-of-frame conditions in compliance with SDH and SONET standards [8]. The states and transitions of this machine are shown in Figure 12. The notation presented there will be

used in the description to follow. All state transitions are synchronized to the falling edge of the system clock, and output lines do not directly reflect the state variables.



FP = frame pulse detected
FP ' = frame pulse detected at proper time
~ FP ' = frame pulse not detected at proper time

Figure 12: Frame Control State Machine

The decision conditions are as follows. On reset, the state machine is forced into the Reset state, which it exits on the next clock edge into the lowest out-of-frame state (OOF 0). This transition is accompanied by a falling edge on the ~SYN line, communicating to the 16:32 demux block and the external 1:16 demux system that word realignment is appropriate. As soon as a frame pulse is detected, the machine moves to the second OOF state (OOF 1), raises the ~SYN line, and begins counting words toward the next frame. When the count reaches 9720, the number of 32 bit words in a frame, the next frame pulse

should arrive if the first was really valid. If a frame pulse is detected at this time (condition FP'), the state machine moves into the first in-frame state (IF 0) and deasserts the OOF line. At this point the SORCC is considered to be in the in-frame state. If instead the proper time comes but no frame pulse is detected (condition ~ FP'), the state machine falls back into OOF 0, dropping the ~SYN line and beginning the frame detection process anew. Once the in-frame state has been reached, several frame pulses in a row must fail to appear on time before the SORCC will reenter the out-of-frame state and restart the frame synchronization process, so there are several states leading from IF 0 back to OOF 0.

The frame control block exports some signals to other blocks based on its knowledge of the frame structure. The LOF/LOS control block receives a signal that indicates either the arrival of the first frame pulse in a new alignment (OOF 0 -> OOF 1) or the arrival of a frame pulse at the proper time (FP'). Thus the signal is asserted every time a frame pulse has been detected and deemed valid, and can be used to meet the requirements for LOS state declaration. The frame control block also exports the SORCC internal frame pulse, aligned to the first A1A1A1A1 word of each frame. It creates this pulse from the counter it uses to determine when to expect the next input frame pulse. Since it is known that input pulses arrives aligned with the third A2A2A2A2 word, a pulse generated 14 words before an input frame pulse's arrival should be aligned to the first word of the frame. This retiming of the frame pulse prevents any improperly timed frame pulses appearing at the SORCC input from propagating through the entire system. The frame control block generates its first output pulse over 9000 clock cycles after detecting the first input pulse, so for IC test purposes its filtering action can be disabled by setting a bit in a register of the microprocessor interface. This causes all incoming frame pulses to be declared valid and passed immediately to the output of the block.

The standards are quite specific in their description of the behavior of the framing circuitry, specifying the required probabilities for false framing and improper loss of frame.

The characteristics of this frame control system will be derived below. As previously mentioned, the first requirement of the standards is that some subset of the A1 and A2 overhead bytes be used to detect incoming frames [8]. In this receiver, the choice of a subset is the responsibility of the frame detect subsystem, but since most currently available options use the bytes 'A1A2A2' for this purpose, the SORCC is designed to work with a 24 bit pattern. The SDH standard specifies that false declaration of the in-frame state should occur with a probability of less than 10^{-5} per 250 μs interval, but that a flawless frame sequence should lead to the declaration of in-frame within 250 μs. The SONET standard does not specify a probability requirement for false framing, but echoes the 250 μs to-frame requirement. Given that the A1A2A2 pattern is used to detect frames, the SORCC's frame control state machine behaves as follows:

Time to frame: $< \approx 250.0$ μs

Just over 125 μs if the frame sequence starts at the beginning of a frame, 250 μs ± a few ns if the frame sequence begins with the A2s

Probability of falsely declaring frame alignment: $<< 10^{-5}$

Appendix 3 discusses the difficulty of specifying this probability exactly and presents calculations that establish an upper bound for it.

The standards also specify that a 10^{-3} bit error rate (BER) should not cause a false OOF more than once per six minutes. Since it is impossible to absolutely guarantee this performance, and the standards fail to specify a probability requirement, one must assume that a system for which the average performance is better than this would be acceptable (i.e. a system for which the expected arrival rate of false OOFs with a 10^{-3} bit error rate signal < 1). It is also specified that OOF should be declared in the absence of framing patterns in not more than 625 μs. The performance of the SORCC's frame control block is as follows:

Time to OOF in absence of frame pulses: $< \approx 625.0$ μs

Just over 500 μs if the frame sequence ends just before a frame pulse was expected, 650 μs ± a few ns if the frame sequence ends just after a frame pulse.

Expected OOFs per 6 min. given a 10^{-3} BER signal: 0.022

10^{-3} BER --> Probability a bit is errored = 10^{-3}

Probability of flawless A1A2A2 string = $(0.999)^{24}$ = 0.976

Probability A1A2A2 string is errored = 1 - 0.976 = 0.0237

Probability of receiving 5 consecutive errored framing patterns = $(0.0237)^5 = 7.5 * 10^{-9}$

Frames in 6 min.: 8000 frames/sec * 60 sec/min * 6 min = 2,880,000

Expected OOF in 6 min.: $(2.88 * 7.5) * 10^{-3}$ = 0.022

So the performance of this framing system meets the requirements of the standards. The specifications described are referenced in [8].

4. Loss-of-Frame / Loss-of-Signal Control

The LOF/LOS control block tracks these two conditions independently. The LOS circuitry uses a rising-edge flip-flop to declare loss-of-signal by asserting the LOS pin as soon as the LOSI input signal is asserted. The clock recovery or frame detection subsystem is responsible for asserting that signal whenever an absence of transitions in the SONET signal longer than 2.3 μs occurs. This is a reasonable partitioning of the task, because those systems must also perform other actions based on a lack of signal. The decision to exit the loss-of-signal state is made by the LOS part of the LOF/LOS control block. It uses a counter driven by the valid frame pulse line from the frame control block and reset by the LOSI line to detect the arrival of two valid frame pulses without intervening assertions of the LOSI line, a condition that corresponds to the specification for exiting the LOS state [15]. When that condition is met, the LOS flip-flop is reset, and the LOS line is deasserted.

The LOF control circuitry consists of three counters. A fast eight-bit parallel counter divides the 77 MHz input clock down to a 303 kHz clock. That clock then drives two 10-bit ripple counters, the OOF counter and the IF counter. The latter counter runs whenever the system is operating in the in-frame state, with the OOF line deasserted, and is reset whenever the system goes out of frame. The former runs whenever the chip is out of frame, and is reset when the IF counter reaches 912. If it reaches 911, the LOF line is asserted,

and stays high until the IF counter reaches 912. The counter values 911 and 912 correspond to cycles of the 303 kHz clock, 3.00 ms in each case. The net effect of the interactions of these counters is an integrating timer which causes the LOF state to be entered if the SORCC accumulates 3 ms of time in the out-of-frame state without ever staying in-frame for 3 ms. It takes 3 ms of uninterrupted in-frame operation to then exit the LOF state. This meets the Bellcore specification for LOF [14].

The LOF block offers a selectable test configuration to simplify IC testing. In this configuration all three counters run off the system clock and are always enabled, and the LOF pin's value is set to the exclusive or of the 303 kHz clock and the LOF value computed from the OOF and IF counters. This results in a square-wave output with a base frequency of 1/256th of the system clock, with a one-system-clock inversion every 1024 cycles, and allows the LOF circuitry to be tested in the course of 1000 test vectors rather than the 500000 or so it would take otherwise.

5. Descrambler

The descrambler implements a parallel equivalent of the seven-bit shift register scrambler described in the standards [9]. The parallel implementation was derived using the state vector methodology described in an AT&T Technical Journal report [16]. Using the state vector notation developed there, the SONET scrambler can be represented by the following transformation matrix and starting state:

$S^T(7) = [\ C0\ C1\ C2\ C3\ C4\ C5\ C6\ ]$ (seven state bits)

$S_0{}^T(7) = [\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ ]$ (start state defined as 1111111)

$S_{n+1}(7) = TR_7\ S_n(7)$

$$TR_7 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

To extend this scrambler to a 32-bit parallel implementation, it is first necessary to create an equivalent serial scrambler with a 32-term state vector. This is accomplished by adding 25 registers with no feedback to the end of the shift register chain shown in the standards. The new starting state is the first 32 bits generated by the original scrambler in operation. A new transformation matrix can be derived for this scrambler; it is a 32x32 matrix in which the first seven lines are the same as TR_7 above, filled to the right with zeros, and the next 25 lines continue the diagonal of 1s evident in TR_7. Call it TR_{32}. This matrix again represents a serial means of generating the scrambling sequence, where one state bit is applied to the input stream per cycle. To make the scrambler work in parallel, all 32 bits of the state vector must be applied at the same time, while the next 32 bits of state are computed. Those bits come from the state vector that would be $S_{n+32}(32)$ in the serial scrambler, so the proper transformation matrix must should yield that result for $SP_{n+1}(32)$. This matrix turns out to be $TR_{32}{}^{32}$:

$$S_{n+32}(32) = TR_{32}(TR_{32}(\ldots S_n(32))) = TR_{32}{}^{32}\, S_n(32)$$
$$SP_{n+1}(32) = TRP\, S_n(32), \text{ where } SP_{n+1}(32) = S_{n+32}(32) \text{ --> } TRP_{32} = TR_{32}{}^{32}$$

A Boolean matrix exponentiation program was written to compute TRP_{32}, and other programs verified that the parallel and serial representations of the scrambler generated the same sequence. The starting state vector and transformation matrix implemented in the descrambler are as follows:

$$S_0{}^T(32) = [1\ 0\ 0\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1]$$

$TR_{32}^{32} =$

$TRP_{32} =$

$$\begin{bmatrix}
0&0&1&1&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}$$

The descrambler block contains a 32 bit register that contains the current state vector. This register is asynchronously set to $S_0(32)$ on the first byte of the frame. Beginning in the cycle of the first SONET word after the C1 overhead bytes, the descrambler computes a new state vector in every clock cycle. It realizes the Boolean matrix multiplication, $S_{n+1}(32) = TRP_{32}\ S_n(32)$, by defining the input to each bit of the state vector flip-flop as the result of an exclusive-or operation performed of a set of its output bits. The required sets were extracted from TRP_{32} by performing an exclusive-or for each "1" in a row: $S_{n+1}(32)<0> = S_n(32)<2>$ xor $S_n(32)<3>$ xor $S_n(32)<4>$ xor $S_n(32)<6>$; $S_{n+1}(32)<1> = S_n(32)<0>$ xor $S_n(32)<3>$ xor $S_n(32)<4>$ xor $S_n(32)<5>$ xor $S_n(32)<6>$, etc. When the descrambler block is disabled, the state vector calculation still occurs, but the computed vector is not applied to the STS data. When it is enabled, the block output is the bitwise exclusive-or of its input

and the state vector. Descrambling is automatically disabled during the A1, A2, and C1 overhead bytes, when the standards specify that no scrambling should occur. Since the state vector is always computed, the descrambling performed by this block will be properly synchronized even if enabled mid-frame.

6. B1 Parity calculation

The B1 calculation block contains an eight-bit storage cell in which it computes the current frame's parity and another in which the computed parity from the last frame is stored. On the first word of each frame, the value in the calculation register is moved into the storage register, and the calculation register is set to the proper parity byte for the input word (Parity<7> = Input<31> xor Input<23> xor Input<15> xor Input<7>, etc.). On all other words, the new parity byte is the xor of the old parity byte with the parity byte that matches the current word (Parity<7> = Parity<7> xor Input<31> xor Input<23> xor Input<15> xor Input<7>, etc.). The parity storage byte is connected to tristate drivers that allow it to drive the parity error check bus when addressed by the B1/B2 Check block.

7. SDH/SONET Cross Connect

The cross-connect is realized as a 4x4 matrix of identical byte-wide storage/switching elements, synchronized by central control circuitry . Each of the switching elements embodies one of the slots depicted in Figure 6. The circuitry of single storage element is presented in Figure 13; the structure of the sixteen element switching matrix, in Figure 14.

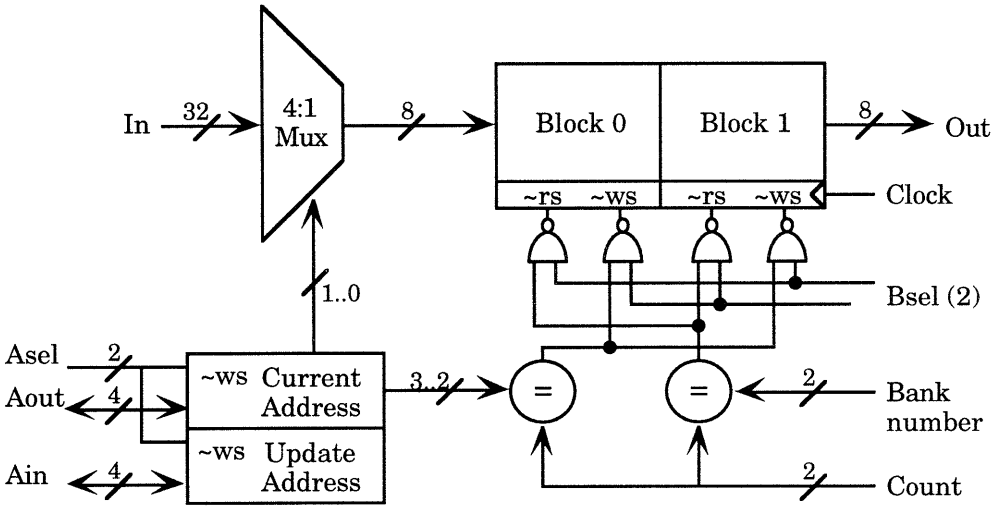


Figure 13: Cross Connect Switching Element

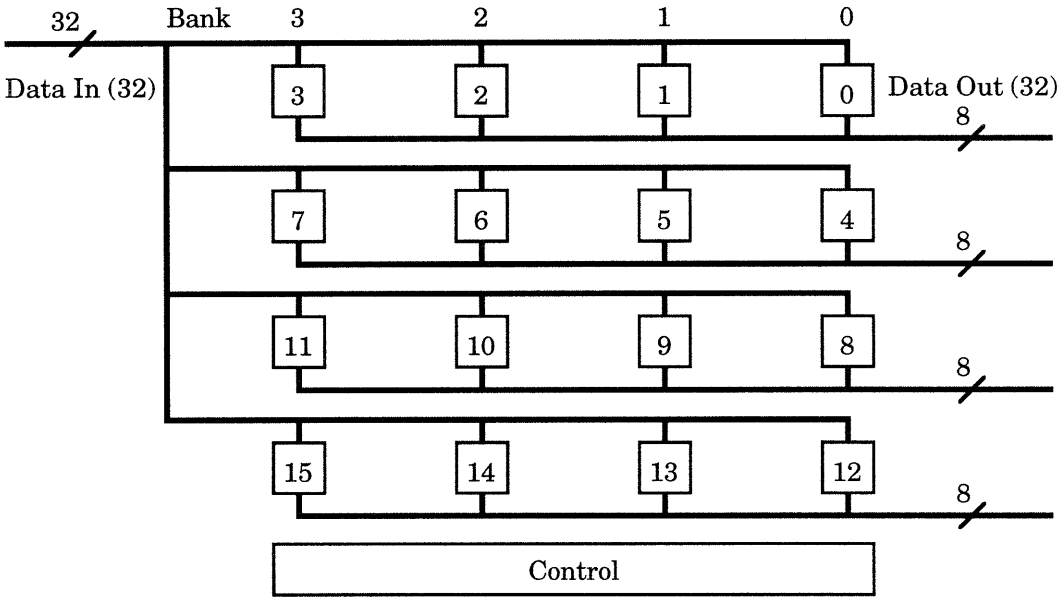


Figure 14: Cross Connect Switching Matrix

Each switching element is made up of two one-byte blocks of data storage, two four-bit address latches, a 4:1 multiplexer, and some control logic. The two bytes of data space are connected in common to an input and a tristate output bus. The element's control logic accepts a pair of block select (BSEL) lines, required to be logical opposites, that determine which block can be written into and which read from at any point in time. Each element has

a built in bank number which it compares to a two-bit word count provided by the central control logic to determine whether it belongs to the bank that should drive the output bus next. If so, one of its data storage blocks will receive a read enable signal (~rs) on the next clock. The word count is also compared to the high two bits of the element's user-defined address, to determine whether that address corresponds to a byte in the next word. If so, one of the data storage blocks will store data on the next clock. The low bits of the address drive the 4:1 multiplexer, selecting which byte of the 32-bit input word should be stored.

An element's user-defined address can be changed through a two-stage process. When the host system wants to reconfigure the cross-connect, it writes the new connection pattern of input STS-3s to output slots into the sixteen half-bytes of slot address space. The connection pattern is created by writing the number of an input STS-3, encoded as four bits, into the address location, half of an addressed byte, corresponding to the output slot it should be routed through. The cross-connect control logic decodes the slot space addresses and sends write select signals (Asel 1) to the appropriate storage elements. Each of the two slots written stores one of the two STS numbers from the parallel system data bus in its "update address" latch (through the Ain lines). The cross connect control logic sets a flag when the last two slots are written at address 17, which causes it to source a transfer enable signal (Asel 2) during the A1 words of the next frame. At that point all 16 elements transfer the address stored in their "update address" latch to their "current address" latch. Since all STS-3s are guaranteed to have the same value during the A1 bytes, this reconfiguration occurs without any loss of data in the output stream.

The switching matrix is organized in four banks of four storage elements. The control logic supplies a 0 to 3 word count synchronized to the frame structure of the input STS-48 to all the elements. This count is used by each element as previously described to determine whether it belongs to the set of cells that are expected to capture or drive data during the next cycle. On every fourth word the block select lines are inverted so that the

data just stored can be read out, and new data can be written over what was just passed downstream. The control logic is also responsible for inserting the proper 5 cycle delay in the internal frame pulse line, decoding the eight addresses that read and write configuration information to the 16 storage elements, generating the transfer enable after the last byte of configuration has been written, and controlling the set of multiplexers and tristate drivers that allow read back of the currently assigned addresses of the storage elements via their address out lines.

The data storage protocol that makes this matrix of storage elements act like a crossbar switch is distributed among them in the form of the built in bank number and the writeable address. The slots are created in the output stream by the rigid structure created by division of the matrix into rows, defined by the connection of elements to the output, and columns, defined by the bank numbers built into the elements. Synchronization to the SONET frame structure allows each element to store only bytes from the STS-3 it addresses. The division of storage into two blocks and the policy of reading from one block while writing to the other turns the combination of controllable storage and structured read back into a pass-through connection scheme. Since each element can store any byte of input data, this connection is nonblocking and best diagrammed as a crossbar.

8. B2 Parity calculations

The B2 calculation block keeps its running parity count in twelve banks of word-wide registers. On each frame pulse, the 48 bytes of parity calculated in the previous frame are transferred from the calculation banks to storage banks. Since the B2 parity does not cover the section overhead, the parity calculation circuitry must be disabled during those words. This is accomplished by setting and clearing an calculation enable bit based on comparisons of constant values to the output of a counter synchronized to the STS-48 frame. Another counter determines which of the twelve banks should reflect the parity information of a given input word. Decoding logic interprets addresses sourced by the B1/B2 check block and

drives the parity error check bus with the storage bank values as appropriate. Since each bit in the parity word only covers one bit in the input word, the parity calculation is simply the exclusive-or of one bit of the accumulated parity word with the same bit of the input word. The parity banks are all reset to zero during the first words of section overhead in each frame, so there is no need to select between methods of parity calculation as in the B1 calculation block.

9. B1/B2 Parity Check

The B1/B2 check block uses a counter synchronized to the input frame structure to determine whether the B1 or one of the B2 bytes is about to appear in the SONET stream. One clock cycle before a given parity word (or partial parity word, in the case of the B1) is expected, the check block drives the address corresponding to that word on the address lines of the parity error check bus. The B1 and B2 calculation blocks decode that address and drive the requested parity word. At the end of that cycle, the requested data is latched into the check pipeline. During the next cycle, a bank of 32 exclusive nors compares the value that appears in the SONET stream to the latched parity word. Bits that do not match generate 1's in the 32-bit output of this comparison. A tree of adders performs a bit addition operation on the comparison word to yield a 6-bit (0 to 32) count of detected errors. The same operation is performed on just eight bits of the comparison word to yield a 4-bit (0 to 8) count of detected errors in the first byte. Both error counts are latched every cycle.

Further logic determines whether the parity value just checked was the B1 byte or one of the B2 words. If the former, only the 4-bit error count applies; if the latter, the 6-bit count is valid. Since a comparison is made in every cycle, there is also a third case, in which the word just checked was neither the B1 nor a B2. In this case neither error count has meaning, and an all-zero count should be used instead. The decision logic controls a one-of-three selector that switches a six-bit version of the appropriate count to the check block's output. The error count block uses the top bit of the address on the parity error check bus to

determine which of its counters to increment, and adds the six-bit output of the check block to that counter every cycle. To prevent the accumulation of spurious "error counts" when the SORCC is in the out-of-frame state, the decision logic drives only zeros on the output when the OOF line is high. The check block is depicted schematically in Figure 15.

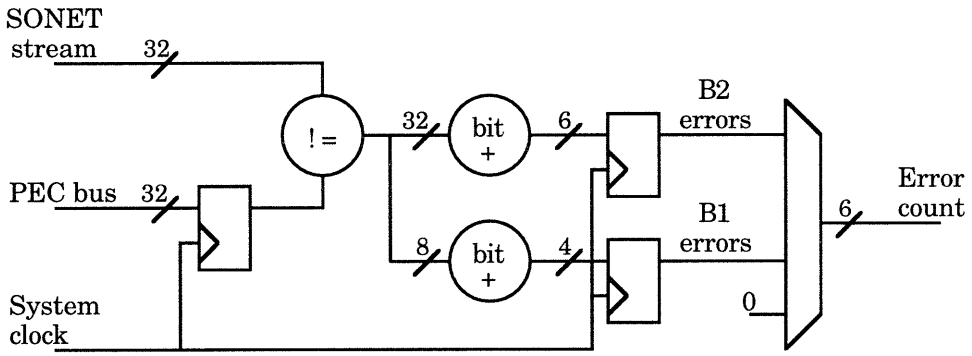


Figure 15: B1/B2 Check Circuit

10. Error Count

The error count block completes the parity error monitoring system. Depending on the value of the high bit of the parity error address bus, the block adds the value of the error count from the B1/B2 check block to its internal one-frame B1 or B2 error counter. The values of these counters are added to the contents of the accumulation registers, copied to the serial output registers, and then cleared at the beginning of each frame. The errored frame accumulator is incremented if any error is detected during a frame. A write operation to the address of a user-visible error count register on the parallel system bus causes the value in the corresponding accumulation register to be latched into that space and then cleared. This protocol provides a stable value in the user-visible counter space while guaranteeing that no errors in the SONET stream go unreported. The error accumulation counters are big enough that they can not overflow if checked often enough: every second for the B1, every 5 seconds for the B2, and every 8 seconds for the errored frame count. The serial output registers are available for reading by the serial overhead output block through the internal serial data bus.

11. Overhead Capture / Storage

The overhead capture and storage block is divided internally into five sections, all of which share some control signals. These sub-blocks are the C1 capture block, which captures the first 16 C1 bytes, the section overhead capture block, which captures the E1, F1, and D1-D3 bytes, the APS channel capture and filter block, which captures and filters the K1 and K2 byte values, the line overhead capture block, which captures the D4-D12, S1, M1, and E2 bytes, and the user-defined capture block. These divisions were introduced to provide a sensible distribution of loads on the parallel system bus address and data lines. Figure 16 shows these sections and their major interface signals. Each section contains a set of capture cells where the actual overhead interception and storage takes place, some decoding logic to determine if one of its memory locations is being addressed on the serial data interface bus or parallel system bus, and some multiplexers and tristate drivers to respond if appropriate. All of the sections use one of two capture cell designs.

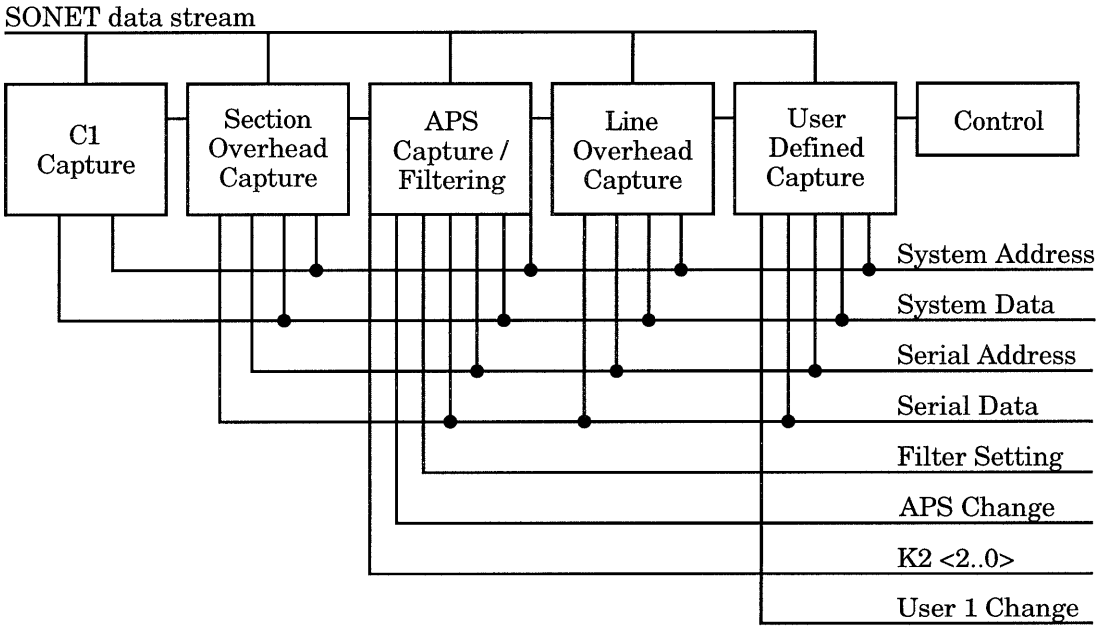


Figure 16: Subdivisions of the Overhead Capture and Storage Block

The dedicated address capture cell is shown in Figure 17; all of the capture functions except user-defined capture are synthesized from these cells. Each cell is created with a pre-

defined address. That address is used at the time of synthesis to connect the cell's input to the proper byte of the SONET word, and to build comparison logic that will cause the cell to latch when the block-global word counter specifies the word to which its address belongs. The cell has two bytes of storage, one to hold the data captured in each frame, and one to hold the data being presented to the user. A transfer signal moves overhead from the capture space to the display space. The force-store signal is for use in testing the SORCC. There is also a force-transfer test signal; it goes through the control circuitry.

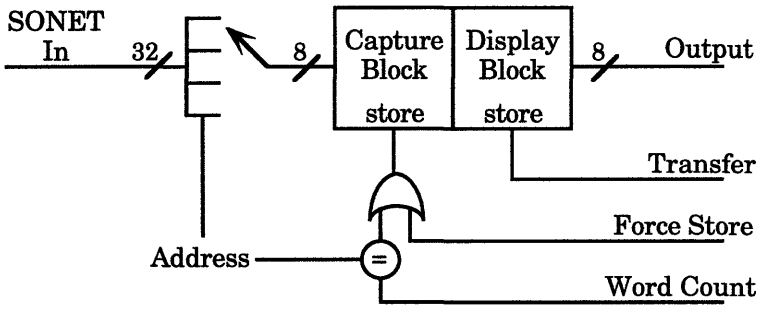


Figure 17: Dedicated Address Capture Cell

The user-defined capture sub-block uses the second cell design, shown in Figure 18. This design differs from the dedicated capture cell in that it has a programmable address. instead of being permanently connected to one byte of the input word, a 4:1 multiplexer driven by the low bits of the address selects the proper byte of the input words. The upper 14 bits of the address are compared to the word count to determine when to latch the data. The shared control circuitry decodes addresses from the parallel system bus and generates write strobes for reprogramming the address of the cells as necessary. The two address bytes themselves come in on the parallel system bus. One level of multiplexing is performed inside the capture cell, driven by externally generated enables, to provide a byte-wide output channel for data or address readout. Since the serial data interface bus always needs to have access to the display data, even if the multiplexer is being used for address read back, a "peek" bus is provided. To provide the option of triggering the snapshot circuitry on a

change in user byte 1, this "peek" also extends to the captured data, so a change in user byte one can be detected by comparing the two peek bytes.

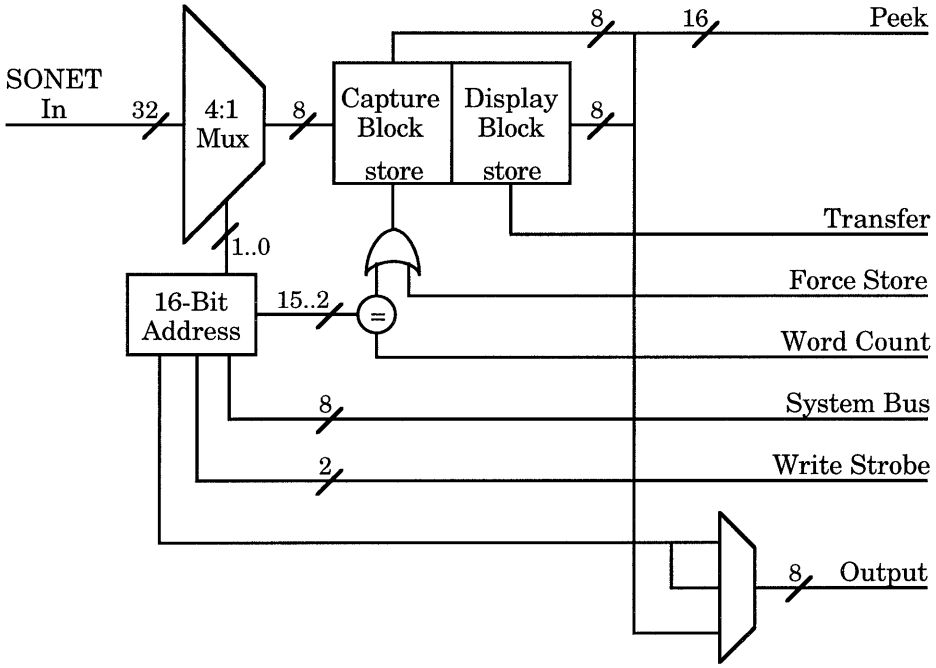


Figure 18: Programmable Address Capture Cell

The overall operation of the overhead capture block requires some coordination among the subsections. The control circuitry generates a word count synchronized to the frame such that all the capture cells can latch data as previously described. It also generates the block-global transfer signal at the end of each frame if overhead update is enabled by the snapshot circuitry. This causes all of the cells to move captured data to presentation space. The force-store and force-transfer test signals also go through the control circuitry for synchronization and routing to all the cells. System and serial address decoding and tristate output bus driving are the responsibility of the individual sub-blocks.

The APS and user blocks contain some functions in addition to capture and read back. The user block sources a trigger signal for the snapshot circuitry whenever the just-captured value of the U1 byte is not the same as its display value. The APS block contains filter circuitry that keeps track of the accepted K1/K2 value, the most recently received

K1/K2 value, and the user-visible K1/K2 value. A received value becomes an accepted value if it is received in 3 or 5 consecutive frames (a control signal from the microprocessor interface specifies which). The user-visible K1/K2 values may differ from the accepted values just as the capture and display blocks of a normal capture cell may differ. The APS subsection exports the low three bits of the accepted K2 value to the AIS/FERF detect block, and generates a signal for the interrupt circuitry whenever the accepted value of the APS channel does not match the user-visible value (i.e., whenever the value of the APS channel changes). The filtered APS values are held constant when the IC is out-of-frame, so no spurious AIS, FERF, or interrupt signals will occur due to IC startup or resynchronization.

12. Serial Overhead Output

The serial overhead block has two major functions: clock generation and data output control. It synthesizes four clocks synchronized to the serial output channels from the 77 MHz system clock. The 576 kHz, 192 kHz, and 64 kHz clocks are all divided out directly from the 77 MHz clock using terminal-count resettable counters, though they all require using both edges of the clock to achieve a 50% duty cycle. Since serial data is to be asserted on the falling edges of these clocks, they all have a falling edge at the beginning of each frame. The 576 kHz clock is generated first. On the falling edge that latches the incoming frame pulse, a two-bit state register is cleared, and a counter is started. 135 edges of the 77 MHz clock later, on a rising edge, one of the state bits is inverted. 135 edges after that, the other bit is inverted. From then on, every 135 rising edges the first bit is inverted, every 135 falling edges the second bit is inverted. The 576 kHz clock is the exclusive-or of the two state bits. The 192 kHz clock is generated similarly from the 576 kHz clock. Its state is cleared by the frame pulse, and then a bit is inverted every three edges of the 576 kHz clock. The 64 kHz clock is generated from the 192 kHz clock in exactly the same manner. As long as the length of the frame corresponds to an even number of cycles of these clocks (it does),

resetting the state to zero on later frame pulses will be invisible in the output clock. Figure 19 illustrates the process.

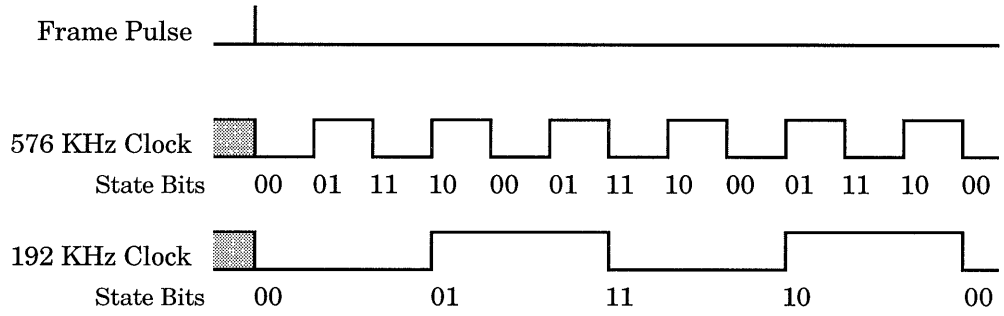


Figure 19: Clock Generation in the Serial Overhead Output Block

The clock for the serial output of the user defined capture bytes (SUCHCLK) must have 192 cycles per frame to accompany the 192 bits. This corresponds to a clock frequency of 1.536 MHz, which cannot be divided out of a 77.76 MHz clock. The closest match is one cycle of the user channel clock per 50 cycles of the system clock. The user clock is disabled for two of its cycles at the beginning of each frame to maintain the proper number of cycles per frame. The result is the gapped 1.5552 MHz serial user-defined channel clock.

The clocks of the serial block drive eight output channels. The section and line overhead and user channel bytes (E1, E2, F1) are streamed out on three channels at a rate of one byte per frame (64 kb/s) per channel. The section data communication channel (D1 - D3) and parity error counts are streamed out on two channels at three bytes per frame (192 kb/s) per channel. The line data communication channel (D4 - D12) is streamed out at nine bytes per frame (576 kb/s). The user capture channel is streamed out at 24 bytes per frame (1.536 Mb/s). For each of these output pipes there is an 8-bit shift register in the serial output block driven by the falling edges of the appropriate serial clock. Each register is loaded in parallel from the serial data bus on the falling edge after the last bit of a byte.

A control-token sequence machine running off the system clock accomplishes the loading of the shift registers. With each of the three fast serial clocks is associated a four-bit

"bit" counter and a "byte" counter of appropriate size. There is a one-bit flag associated with the 64 kHz clock. Each "bit" counter counts the falling edges of its clock. A sequence begins whenever any of the "bit" counters reaches 8 or when the slow channel's flag is set. The serial overhead output block's operation for a frame is as follows: the frame pulse clears all of the byte counters, sets all of the "bit" counters to 8, and sets the 64 kb/s channel's flag bit. This creates a start condition for the sequence machine.

The start of the sequence sets a flag that removes the start condition, preventing a second sequence process from being started while the first is still active. The control machine then goes through a set of steps for each of the serial output clocks, starting with the fastest. For each, it checks to see if the highest bit of the corresponding "bit" counter is set (or if the flag is set for the 64 kb/s clock). If so, it goes through a sequence of steps for each channel associated with that clock. First it drives the sum of the defined base address for the channel and the "byte" counter value onto the serial data interface bus address lines. The requested data appears on the data lines of that bus within about 15 ns, so at the end of the next cycle of the system clock the control machine asynchronously loads the channel's shift register with whatever value is on the bus. These two steps repeat for all channels associated with the clock, then the "byte" count is incremented and the "bit" count is cleared. After all channels have been checked, and loaded if necessary, the flag denoting a sequence in progress is cleared.

Since each step in the sequence machine's operation takes one cycle of the 77 MHz clock, all of this happens very quickly on the time scale of the output channels. Right after the frame pulse, the machine will find all bit counters set high and load all of the shift registers. After that, whenever a channel's bit counter reaches 8, it triggers a new pass through the sequence machine. At that time all channels that need loading will be loaded with the next byte from their address space, and the serial output operation will continue. Allowing one sequence machine to orchestrate the loading of all output pipes prevents

contention on the serial data interface bus. A flow diagram of the control sequence machine is shown in Figure 20.

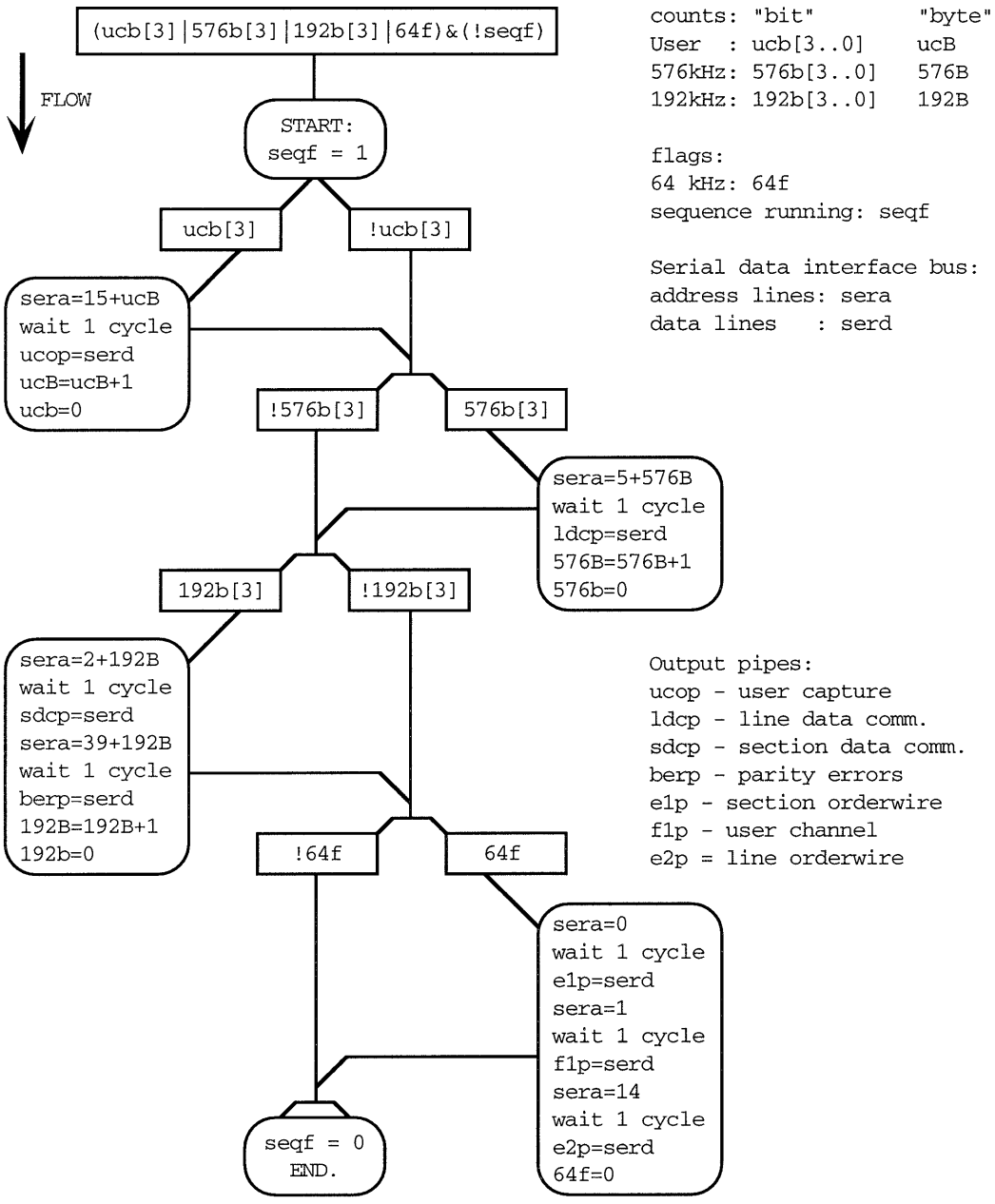


Figure 20: Control Flow in the Serial Output Sequence Machine

13. Line Alarm Indication Signal Insertion

The line AIS insertion block takes replaces most the SONET stream with an all-ones pattern when activated via an external pin (LAIS). The only part of the stream left untouched is the section overhead, as the standards specify [11]. The synchronization circuitry for this function is borrowed from the B2 calculation block: a control flag is toggled high and low according to the value of a frame-aligned counter. The logical "and" of this flag and the value of the LAIS pin drives a word-wide 2:1 multiplexer that selects for its output either the input SONET stream or a word comprised entirely of ones.

14. Scrambler

The scrambler was derived using the same methodology described in the discussion of the descrambler. Instead of one 32-bit parallel sequence generator, the scrambler block creates a single 8-bit-wide parallel sequence generator, and applies the eight state bits to each of the four bytes of the SONET data word. This in effect forces the data stream into the format of four STS-12s: four scrambled one-byte channels running at 77.76 MHz is equivalent to four one-bit, serially scrambled channels running at 622.08 Mb/s. As in the descrambler, the generation of new state vectors and the application of old ones begins in the word after the last C1. The implementation is analogous. For the 8-bit parallel sequence generator, the seed state and transformation matrix are:

$S^T(8) = [\ C0\ C1\ C2\ C3\ C4\ C5\ C6\ C7]$

$S_0{}^T(8) = [\ 0\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ ]$

$SP_{n+1}(8) = TRP_8\ SP_n(8)$

$$TRP_8 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

15. Microprocessor Interface

The four principle functions of the microprocessor interface were listed previously: exerting control through the master control register, monitoring system signals and sourcing interrupts based on them, running the snapshot circuitry that enables and disables overhead update, and creating data acknowledgment signals during read and write accesses. The master control register contains eight bits, defined as follows. Bits 7 and 6 specify the position of the incoming frame pulse with respect to the A1A2 boundary in the incoming frames. These bits connect to the frame pulse delay block. Bit 5 sets the number of times in a row the same value must be received in the K1 or K2 byte before it is accepted as the new value for that byte. If it is set to one, the value will be accepted on the fifth reception [13], if zero, on the third [13']. Bit four is an on/off "switch" for the descrambler. If it is high, the descrambler is activated. Bit three performs the same function for the scrambler. Bits two and one control the trigger event of the snapshot decision circuitry. Their function will be described later. Bit 0 allows a chip-level reset to be performed without actually using the dedicated ~RESET pin. The SORCC should always be reset on power-up; it might also prove useful to reset it to restore default settings at other times. If bit 0 is written low, or the dedicated ~RESET pin is driven low, a chip-level reset will be performed. Timing is performed automatically for the master control register reset; if the dedicated pin is used, it should be held low for at least 100 ns. The default settings of the IC are presented in the address space descriptions in appendix 2.

The interrupt control circuitry communicates with the system through the interrupt mask register (IMR), the interrupt status register (ISR), the interrupt pin, and the interrupt acknowledge pin. An interrupt is sourced when the interrupt (~INT) pin is driven low by the interrupt control circuitry, and is pending until the control system acknowledges it by lowering the interrupt acknowledge (~IACK) pin. When an interrupt has been acknowledged, the ~INT pin is released and the ISR is cleared for one clock cycle. A new

interrupt could be sourced immediately after that. Eight conditions are monitored: AIS, FERF, OOF, LOF, LOS, APS change, snapshot, and frame. The first five of these conditions are determined by the dedicated outputs of the same name. The APS change signal indicates that a new APS value has just been accepted by the filtering circuitry of the overhead capture block's APS subsystem, and will be readable after the next frame pulse (or immediately, through the unfiltered K1 and K2 registers). The snapshot signal indicates that overhead data update has stopped; that is, that the data made visible by the capture block will not change until the snapshot option is reset. The frame signal indicates that a new frame has begun. For each of these signals, there is a bit in each interrupt register. The IMR determines which of these signals may cause an interrupt. If a signal's bit is set in the IMR, the SORCC will source an interrupt whenever none is pending and that signal is active (high). The ISR indicates the current value of the monitored signals when no interrupt is pending. When an interrupt is sourced, the ISR locks its version of the value of all those signals for which interrupts are enabled in the IMR. This allows the user to determine exactly what condition caused the interrupt by logically anding the values of the ISR and IMR. The ISR will continue to reflect the current status of signals for which interrupts are disabled. Acknowledging an interrupt does not necessarily clear the condition that caused it, so the controller may wish to adjust the IMR to disable interrupts based on the condition being acknowledged before asserting ~IACK.

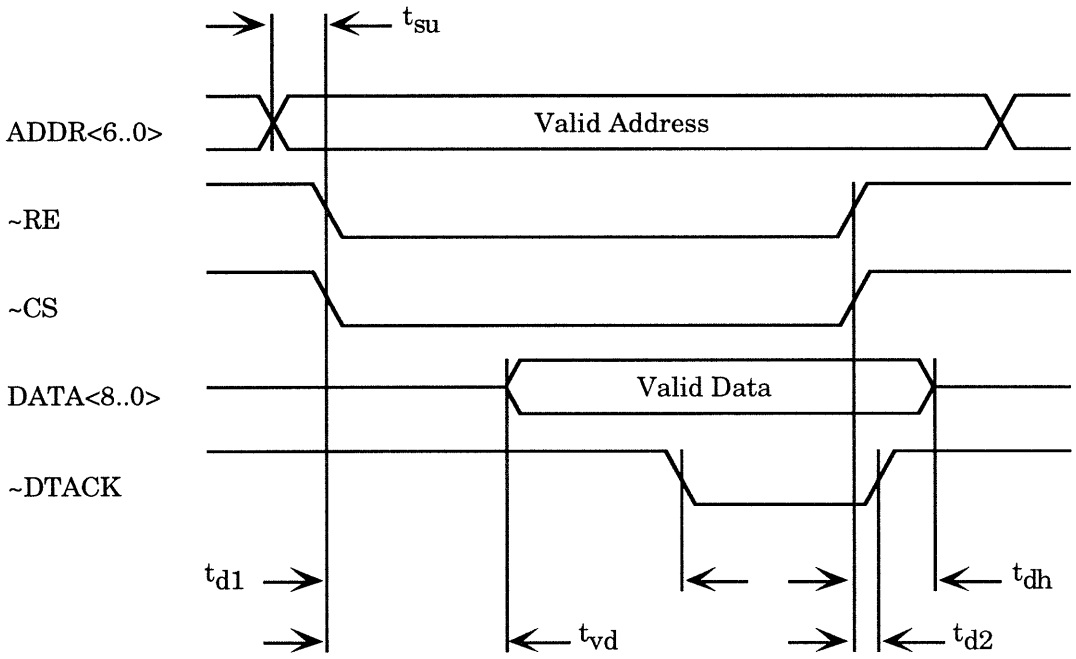
The snapshot decision circuitry reads its setting from bits 2 and 1 of the master control register and from the snapshot activate (SNAP) pin. The SNAP pin overrides any other settings; if it is high, the overhead capture block will not be allowed to update its display cells. This takes effect immediately upon assertion of the pin. The snapshot controls in the master control register determine the behavior of overhead update when the SNAP pin is low. If both bits are low, overhead data will be updated with every new frame. If they are set to "'01", the current frame's overhead will be transferred to visible space with the next frame pulse, then update will be inhibited. If they're set to "10", update will cease

after the next frame in which the value of the first user-selectable capture byte changes. This option allows the host system to use any byte in the frame as a trigger for the capture of a frame's overhead. If the control bits are set to "11", overhead update will be blocked after the next frame during which an interrupt is sourced. This allows capture of the overhead of any frame that causes AIS, FERF, etc. Resetting the bits to "00" will re-initiate update after any of the other settings halts it. Since the AIS/FERF detect block receives a signal based on the currently accepted value of the K2 byte rather than its user-visible value, its operation is not compromised by activation of the snapshot circuitry.

The data acknowledge (~DTACK) signal exists to allow the SORCC to interface with Motorola-type microprocessors and microcontrollers. When either a read or a write is begun by lowering chip select (~CS) and read enable (~RE), or by lowering ~CS and write enable (~WE) and applying valid data on the parallel system bus, a timing chain in the SORCC is activated. Three system clock edges later, the ~DTACK line is lowered, indicating that the transaction is ready to be completed. At that point the processor should raise the ~CS and ~RE or ~WE lines. In a write, data must remain valid until just after ~CS and ~WE go high. The ~DTACK signal has nothing to do with how data is actually read and written in the internal registers, it just supplies timing cues to the microprocessor that are appropriate to those operations.

Much mention has been made of accessing user readable and writeable space through the parallel system bus. A map of that space is available in appendix 2, and the operations that access it are described here. Although the physical memory within the SORCC is distributed, the microprocessor interface was assigned the responsibility for codifying the strategy for accessing it. An example read access proceeds as follows. The host system drives the desired address onto the address lines of the SORCC parallel system bus, then activates the ~RE and ~CS lines. In the case of a read access, it is acceptable for both these lines to drop immediately with the address. About 20 ns later, the data requested

appears on the data lines of the bus. At that point the controller may latch the data, raise the ~RE and ~CS lines, and change the address. If the controller uses the ~DTACK interface, it will wait until the ~DTACK signal is lowered by the SORCC (about 24 ns after both ~CS and ~RE go low) to latch the data and raise the control lines. A read access is shown in Figure 21. The timing specifications given are estimates based on preliminary simulation.

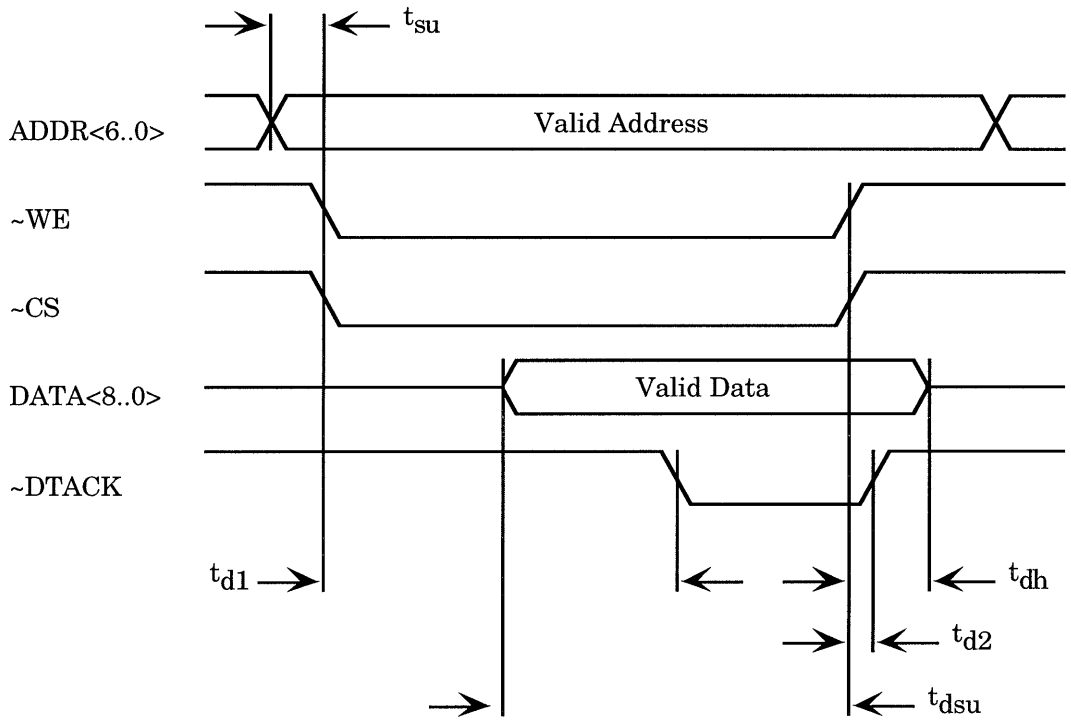


Read Cycle: t_{su} = 0 ns $t_{d1} \approx$ 24-32 ns $t_{d2} \approx$ 6 ns $t_{vd} \approx$ 20 ns $t_{dh} \approx$ 4 ns

Figure 21: Read Access Timing

A write access is similar. The controller asserts an address, then lowers the ~WE and ~CS lines. In this case, a setup time of about 6 ns is required to ensure that only the proper address will be written. It is acceptable for one but not both of the control lines to be asserted before this time. Then the controller should assert the data to be written on the data bus. 20 ns later, the data will be stable in the proper latch in the SORCC, so the control lines can be deasserted. The data must be held for about 4 ns after the first control line is raised. Again, a system that uses the ~DTACK line would wait for that line to drop

before releasing its control signals. The same overall data window and hold time requirements apply. Figure 22 is a timing diagram for a write cycle. Again, the specifications shown are based on simulation data.



Write Cycle: t_{su} = 6 ns t_{d1} = 24-32 ns t_{d2} = 6 ns t_{dsu}= 20 ns t_{dh} = 4 ns

Figure 22: Write Access Timing

16. Alarm Indication Signal / Far End Receive Failure Detection

The AIS/FERF detection block simply monitors the three bits created for it by the APS subsection of the overhead capture block, which correspond to the low bits of the currently accepted K2 byte. If they contain the alarm indication signal code, "111," the AIS pin is raised. If they contain the far end receive failure code, "110," the FERF pin is raised. If neither of these values is encoded, both the AIS and FERF pins are lowered. The outputs of this block are latched to prevent glitches.

IV. Conclusion

The architecture and design presented in the previous sections specify a processor that implements the framing and de-interleaving functions necessary to recover four 622 Mb/s SDH/SONET streams, output on byte-wide 77.76 MB/s channels, from a 2.488 Gb/s SDH/SONET input stream, entering on a 16-bit channel at 155.52 Mb/s per line. These specifications have been encoded in a Tektronix proprietary hardware description language, compiled to a netlist of standard cells from the Tektronix CMOS library, and subjected to extensive simulation with library default parasitic capacitance values. The simulation results indicate clearly that the overall design is sound.

Before the SORCC can be fabricated as an integrated circuit, it must also pass through an iterative layout loop. In that process the elements and connections specified by the netlist are placed and routed according to the geometric constraints of the target CMOS fabrication process to create a layout for the IC. Accurate parasitic capacitance estimates are extracted from that layout, and further simulation of the circuit is performed using those values rather than library defaults. The design is then modified to correct any indicated loading or timing problems, and the process repeats. Since the place and route operation and the parasitic capacitance extraction are well automated, each iteration of this process is relatively short (limited by simulation and design modification time). Once simulation indicates that the design and layout will work together, at the required speed and over expected fabrication process variations, the layout is sent out for fabrication.

The SORCC design is currently going through this layout process. Because of the relatively high clock rate of the circuit, some tweaking of buffer strengths and rearrangement of heavily loaded nodes is necessary to bring the simulated circuits up to the required speed. The most recent layout and design are very close to acceptable, so it is expected that the SORCC will be in fabrication by the time this thesis is published.

Appendix 1: Input/Output Signals of the SORCC

There are a total of 113 signal pins defined in the SORCC design. The following tables summarize their names, types and functions.

Global Control, Alarm, and Synchronization Signals		
Pin Name	Pin Type	Function
~RESET	TTL Input	The active low reset signal forces the internal state of the IC to its default setting. This will cause the OOF state to be entered.
SNAP	TTL Input	When this input is low, each frame's overhead may be captured and available for reading during the next frame as set in the internal registers. When it is high, the update of the overhead capture registers is inhibited.
LAIS	TTL Input	When this input is asserted high, the line alarm indication signal block will insert line AIS in the SONET stream(s).
LOSI	TTL Input	The active high LOSI line should be asserted by the clock or frame detect system when no transitions have been detected in the input stream for at least 2.3 μs.
FPO	TTL Output	The active high frame pulse output will be asserted along with the first A1 byte of each output frame (on SDO[1..4]). It provides frame synchronization information for the STS-12 outputs and all serial overhead outputs
OOF	TTL Output	The active high out-of-frame signal will be asserted when the IC is in the out-of-frame state.
LOF	TTL Output	The active high loss-of-frame signal will be asserted when the loss-of-frame state occurs. (Bellcore 6.3.1.1.2, with integrating timer as per 6-133)
LOS	TTL Output	The active high loss-of-signal line will be asserted when the IC is in the loss-of-signal state.
AIS	TTL Output	The active high AIS alarm signal is asserted when the filtered value of the captured K2 byte holds the AIS code in bits 6-8. (111)
FERF	TTL Output	The active high FERF alarm signal is asserted when the filtered value of the captured K2 byte holds the FERF code in bits 6-8. (110)

SONET STS-48 input		
Pin Name	Pin Type	Function
ICLK	Differential ECL Input	The 77.76 MHz, nominally 50% duty cycle, input clock must be synchronized to the SDI input signals. SDI<15..0> are sampled on both edges of this clock
~SYN	TTL Output	A falling edge on the synchronization signal should cause the frame detect system to perform word realignment.
FPI	Differential ECL Input	The active high frame pulse input should be asserted for one ICLK cycle per frame detected, between the first and fourth A2A2 words.
SDI<15..0>	Differential ECL Inputs	The synchronous data input bus receives the STS-48 stream in 16 bit word-serial format. SDI<15> is the most significant bit (MSb), corresponding to the first bit sent in a serial SONET stream. The word should be aligned such that the first A1 byte falls into SDI<15..8>.

STS-12 output		
Pin Name	Pin Type	Function
SDOCLK	TTL Output	The 77.76 MHz synchronous data output clock provides timing for the four SDO channels. SDO[1..4]<7..0> are asserted on the falling edge of this clock.
SDO1<7..0>	TTL Output	The synchronous data output bus number 1 carries STS-12 stream number 1 in byte serial format. SDO1<7> is the MSb.
SDO2<7..0>	TTL Output	The synchronous data output bus number 2 carries STS-12 stream number 2 in byte serial format. SDO2<7> is the MSb.
SDO3<7..0>	TTL Output	The synchronous data output bus number 3 carries STS-12 stream number 3 in byte serial format. SDO3<7> is the MSb.
SDO4<7..0>	TTL Output	The synchronous data output bus number 4 carries STS-12 stream number 4 in byte serial format. SDO4<7> is the MSb.

Microprocessor Interface		
Pin Name	Pin Type	Function
~CS	TTL Input	The active low chip select enables the internal registers of the IC to be read or written
~RE	TTL Input	The active low read enable causes the contents of the register specified by ADDR<7..0> to be driven on DATA<7..0> iff ~CS is asserted.
~WE	TTL Input	The active low write enable causes the data driven on DATA<7..0> to be written into the register specified by ADDR<6..0> iff that register is writeable and ~CS is asserted. Data is latched on the rising edge of ~WE/CS.
~DTACK	TTL Output Open Drain	The active low data acknowledge pin will be asserted during read and write accesses to provide an interface for 68000-style processors. This pin floats when not asserted, allowing wire-and operation. A 1KΩ or greater external pull-up is required.
ADDR<6..0>	TTL Input	The value driven on the 7 bit address bus specifies which of the internal registers should be read or written.
DATA<7..0>	TTL Input/Output	The tristate bidirectional data bus conveys data to or from the register specified by ADDR<6..0> as specified by ~RE, ~WE, and ~CS.
~INT	TTL Output Open Drain	The active low interrupt pin signals error and changed states to the chip's controller. This pin floats when not asserted, allowing wire-and operation. A 1KΩ or greater external pull-up is required.
~IACK	TTL Input	The active low interrupt acknowledge resets the interrupt state, causing ~INT to be released.

Serial Overhead Outputs		
Pin Name	Pin Type	Function
64KCLK	TTL Output	The 64 kHz clock provides timing for the serial channels SE1, SE2, and SF1. Data is asserted on the falling edge of this clock. The FPO signals the start of a new byte.
SE1	TTL Output	The serial E1 line transmits the section orderwire byte captured from the previous frame in a bit-serial fashion, MSb first.
SE2	TTL Output	The serial E2 line transmits the line orderwire byte captured from the previous frame in a bit-serial fashion, MSb first.
SF1	TTL Output	The serial F1 line transmits the section user channel byte captured from the previous frame in a bit-serial fashion, MSb first.
192KCLK	TTL Output	The 192 kHz clock provides timing for the SSDC and BERR channels. Data is asserted on the clock's falling edge. The FPO signals the start of a new 3-byte string on the SSDC and BERR lines.
SSDC	TTL Output	The serial section data com. line transmits the three section data com. bytes (D1-3) captured from the previous frame in a bit-serial fashion.
BERR	TTL Output	The bip error output line transmits the count of errors detected in the frame before the previous. The first byte is the B1 error count (0-8), the second and third are the high and low bytes of the B2 error count (0-384) . These bytes are transmitted bit-serially, MSb first.
576KCLK	TTL Output	The 576 kHz clock provides timing for the SLDC serial overhead channel. Data is asserted on the falling edge of this clock. The FPO signals the start of a new nine-byte string on the SLDC line.
SLDC	TTL Output	The serial line data com. line transmits the nine line data com. bytes (D4-12) captured from the previous frame in a bit-serial fashion.
SUCHCLK	TTL Output	The serial user choice clock is a gapped approx. 1.5 MHz clock that provides timing for the SUCH channel. Data is asserted on the falling edge of the clock. FPO signals the start of a new frame's worth of data on the SUCH line.
SUCH	TTL Output	The serial user choice line transmits the user selectable capture data bytes from the previous frame in a bit-serial fashion, starting with the MSb of byte U1.

Appendix 2: Address Spaces of the SORCC

The SORCC contains three separate address spaces corresponding to its three communication busses. Each space maps values on the address lines of a bus to memory locations that can be read or written through its data lines. The access protocol of the internal unidirectional busses requires the bus master to drive a value on the address lines, and a slave to respond by loading the addressed data onto the data lines. A simple map is enough to clearly specify these spaces. The access protocol of the parallel system bus is in general read/write, but some locations are read-only, some contain control switches rather than simple data, some perform special functions when accessed, and many have "default values" loaded into them on system reset. To clearly specify the behavior of this space, it is important to list not only what is stored at each address, but also what its function is, what its default value is, and how it can be manipulated. Thus the "map" of the parallel system bus address space is rather detailed.

Parity Error Check Bus Address Space:

5 address bits gives 32 addressable locations. Addresses listed in hexadecimal notation.

0	B2 word 0
1	B2 word 1
2	B2 word 2
3	B2 word 3
4	B2 word 4
5	B2 word 5
6	B2 word 6
7	B2 word 7
8	B2 word 8
9	B2 word 9
A	B2 word 10
B	B2 word 11
C - F	All ones bytes
10 -1F	B1 byte (high bit of address high)

Serial Data Interface Bus Address Space:

6 address bits gives 64 addressable locations. Addresses listed in decimal notation.

0	E1	
1	F1	
2 - 4	D1 - D3	(in order)
5 - 13	D4 - D12	(in order)
14	E2	
15 - 38	User defined capture bytes 1 - 24	(in order)
39	B1 errors last frame	
40	B2 errors last frame, high byte	
41	B2 errors last frame, low byte	
43 - 63	unused	

Parallel System Bus Address Space:

7 address bits gives 128 addressable locations; Addresses in binary and decimal notation. Data is read and written through the bidirectional data bus DATA<7..0>, where DATA<7> is the most significant bit (MSb) and DATA<0> the least significant bit (LSb) of each byte accessed. The reset value given (if any) is the value to which the storage location is initialized during chip reset. Bytes that are read/write will hold their reset value until a new one is written into them. Read only bytes will be updated automatically as per their function. Side-effects are noted.

Master control register: Read / Write

Address: 0000000 (0)

Reset value: 10111001 (185)

LSb (0)	1	2	3	4	5	6	MSb (7)
~reset	snap 0	snap 1	scen	dsen	filter	fpp 0	fpp 1

~ reset: (1) Writing a 0 to this bit has the same effect as asserting the ~RESET pin: it forces the internal state of the chip to its default state. This includes restoring default values to all control registers (which restores the ~reset bit to 1). This mode of reset lasts between three and five clock cycles after the write that starts it is concluded. The chip should not be accessed in that time.

snap <1..0> : (00) These bits specify the state of the overhead "snapshot" circuitry. If the SNAP pin is asserted, it overrides the contents of this register. The functions are as follows:

snap 1	snap 0	Function
0	0	Capture every frame's overhead. Re-initiate capture after a snapshot (01 or 10).
0	1	Inhibit frame overhead capture after current frame.
1	0	Capture the overhead of the next frame in which the value of the user defined capture byte number 1 changes.
1	1	Capture overhead of the next frame in which an interrupt occurs.

scen: (1) This is the scrambler enable bit. If it is 1, the STS-12 outputs will be scrambled by the frame-synchronous scrambler. If it is 0, scrambler operation is disabled.

dsen: (1) This is the descrambler enable bit. If it is 1, the incoming STS-48 will be descrambled as it passes through the IC. If 0, the descrambler will be turned off.

filter: (1) This bit switches the filter mode for the APS overhead bytes (K1 and K2). If it is low, the K1 and K2 values will be accepted after the third reception. If high, after the fifth. The AIS and FERF outputs are based on the filtered K1 and K2 values.

fpp <1..0> : (10) These bits specify the input frame pulse position:

fpp 1	fpp 0	Timing
0	0	Along with first A2A2.
0	1	Along with second A2A2.
1	0	Along with third A2A2.
1	1	Along with fourth A2A2.

Interrupt mask register: Read / Write

Address: 0000001 (1)

Reset value: 00000000 (0)

LSb (0)	1	2	3	4	5	6	MSb (7)
AIS	FERF	OOF	LOF	LOS	ΔK1/K2	snap	fp

AIS: If this bit is set, interrupts will be sourced on detected AIS conditions.

FERF: If this bit is set, interrupts will be sourced on detected FERF conditions.

OOF: If this bit is set, interrupts will be sourced on detected out-of-frame conditions.

LOF: If this bit is set, interrupts will be sourced on detected loss-of-frame conditions.

LOS: If this bit is set, interrupts will be sourced on detected loss-of-signal conditions.

ΔK1/K2: If this bit is set, interrupts will be sourced every time the filtered value of the K1 or K2 byte changes.

snap: If this bit is set, an interrupt will be sourced when the requested overhead snapshot has been completed.

fp: If this bit is set, an interrupt will be sourced with each new frame. This lets the user know when new overhead data, etc. is available for access through the bus.

By default, all interrupts are disabled.

Interrupt status register: Read only

Address: 0000010 (2)

Reset value: 00000000 (0)

LSb (0)	1	2	3	4	5	6	MSb (7)
AIS	FERF	OOF	LOF	LOS	ΔK1/K2	snap	fp

This register should be anded with the mask from register 1 to see what condition caused an interrupt. It can also be polled to determine the current state of disabled interrupt conditions. When an interrupt occurs, enabled bits will be held until the interrupt is acknowledged. Other bits will change with the conditions they represent. All bits will be cleared on interrupt acknowledgment, update will then recommence when ~IACK is raised.

Errored frame count registers: Side-effect Write / Read

Address: 0000011 (3), 0000100 (4)

Reset value: 0

Register 3 contains the less significant byte of the errored frame count, register 4 contains the more significant byte. An errored frame is one in which one or more B1 or B2 errors are detected. Internal counters are updated during the first word of the frame with the last frame's error count. Whenever either address 3 or 4 is written, the internal counter is latched into the user-readable count registers and reset. Timing logic ensures that no errors go uncounted, so the latch-and-clear operation may not complete until a few tens of nanoseconds after the write is initiated. This special write access should meet the same timing requirements as a normal write, but the value on the data lines will be ignored. The counter is guaranteed not to overflow if checked at least every eight seconds. If it is not checked within that time, its count may be incorrect.

B1 error count registers: Side-effect Write / Read

Address: 0000101 (5), 0000110 (6)

Reset value: 0

Register 5 contains the less significant byte of the B1 error count, register 6 contains the more significant byte. Internal counters are updated once per frame and transferred to these addresses when any one of them is written. The protocol is the same as described for the errored frame count registers. The internal counter is guaranteed not to overflow if checked at least every second. Otherwise its count may be incorrect.

B2 error count registers: Side-effect Write / Read

Address: 0000111 (7), 0001000 (8), 0001001 (9)

Reset value: 0

Registers 7, 8, and 9 contains the B2 error count, with the least significant byte in 7 and the most significant in 9. These means by which this count may be read are like those described in the previous two sections: a write to 7, 8, or 9 causes a latch and clear operation to be performed, after which the accumulated error value can be read in those addresses. The 24 bit B2 counter is guaranteed not to overflow if checked at least every 5 seconds.

Cross-Connect definition registers: Read/Write

Address: 0001010 through 0010001 (10 through 17)

Reset values (hex): 01 23 45 67 89 ab cd ef

These registers specify how to put the 16 STS-3s that make up the incoming STS-48 into the 16 output slots in the outgoing data stream. The sixteen half bytes held in registers 10 through 17 correspond to the 16 slots in the output stream: slot 0 through 3 are in byte 0 (SDO1), 4 through 7 in byte 1 (SDO2), 8 through 11 in byte 2 (SDO3), and 12 through 15 in byte 3 (SDO4) of the outgoing words. The value in each of the half bytes specifies an STS-3 from the incoming STS-48. Writing an STS-3 number into a slot's half-byte has the effect of setting the cross connect to output that STS-3 through that slot. These registers must be written in order from 10 to 17. After register 17 has been written, the connection pattern will be changed during the A1 bytes of the next frame. Values can be read back in any order at any time. The reset values configure the cross-connect to act as an STS-48 to STS-12 demultiplexer as per SONET and SDH standards. Then SDO1-4 represent four STS-12 streams.

Overhead registers:

Address: 0010010 through 1111110 (18 through 126)

Captured overhead is stored in these registers as permitted by the snap <1..0> bits in the master control register and the external SNAP line. If the external SNAP line is asserted, these registers will not be updated. If the snap<1..0> code dictates a single frame capture, and that capture has occurred, these registers will not be updated until re-enabled by the proper code (snap<1..0> = 00, SNAP = low). If continuous capture is in effect, these registers will be updated during the last word of each frame with the data captured during that frame.

Named overhead: Read only

Address: 0010010 through 0110100 (18 through 52)

Not affected by reset

Registers 18 through 52 contain the most recently captured named overhead bytes. The use of the first 16 C1s for section trace, the first Z1 (renamed S1) for synchronization status, and the third Z2 (renamed M1) for FEBE are under recommendation to the standards bodies and not yet written as standard [5]. The rest of these bytes are the named overhead from the first STS-1 of the STS-48 signal [4]. In order, the captured bytes are:

18-33: sixteen C1 bytes
34: E1 byte (section orderwire)
35: F1 byte (section user channel)
36-38: D1 - D3 (section data com. channel)
39-40: K1, K2 (APS channel, as directly captured)
41-49: D4 - D12 (line data com. channel)
50: S1 byte (first Z1, used for synchronization status)
51: M1 byte (third Z2, used for FEBE)
52: E2 byte (line orderwire)

Filtered overhead: Read Only

Address: 0110101, 0110110 (53, 54)

Reset value: 0

Registers 53 and 54 contain the filtered K1 and K2 values, accepted as defined by the filter bit in the master control register. The AIS and FERF alarm signals are generated

from the K2 value in this register. The filtered values are cleared on reset, and no filtering occurs while the IC is in the OOF state.

User definable capture bytes: Address Read/Write, Value Read Only
Address: 0110111 through 1111110 (55 through 126)
Reset values: given below

Registers 55 through 126 control the user definable capture bytes. The first two bytes of each three-byte group specify the "address" within the STS-48 of the byte to capture; they are read-write. The more significant byte of that address is stored in the byte with the higher address in this space. The third byte is read only and contains the last captured value of the desired byte from the STS-48. There are 24 such groups of three:

55-57, 58-60, 61-63, 64-66, 67-69, 70-72,
73-75, 76-78, 79-81, 82-84, 85-87, 88-90,
91-93, 94-96, 97-99, 100-102, 103-105, 106-108,
109-111, 112-114, 115-117, 118-120, 121-123, 124-126.

The data values in these registers are subject to the same snapshot rules as the other capture registers. Updates to the "address" of these registers will propagate immediately to the capture circuitry (there is no "lull" in which to update them), so at least one frame should be allowed to pass after writing a new address before reading out the captured data.

User definable capture byte number one also serves the special purpose of acting as a trigger for the snapshot circuitry. If snap<1..0> is set to 10, frame overhead will be captured for every frame until the value of byte 57 changes. The frame in which the change occurs will be captured normally. After that a snapshot condition will be recognized and the update of the overhead registers will halt until snap<1..0> is reset to 00.

The "address" of a byte within the frame is simply its byte number as determined by order of transmission. Thus the first A1 is at address 0, the first A2 is at address 48, the B1 is at 4320. A simple scheme for determining the address of a byte is:

address = position in row + 4320 * row number

where "position in row" is between zero and 4319 (inclusive), and the row number is from 0 to 8 (inclusive).

On reset, the two address registers of each user defined capture cell will be set to a default value. Reset has no effect on the data register of each cell. The values are:

U1: 42 (a byte in the A1s, which should not ever change)
U2, U4, U6, U8: 12960 - 12963 (The H1 bytes from the STS-3s of the first STS-12)
U3, U5, U7, U9: 13008 - 13011 (The H2 bytes from the STS-3s of the first STS-12)
U10 - U24: 112 - 126 (The bytes directly after the 16th C1)

Test register: Read / Write
Address: 1111111 (127)
Reset value: 00000000 (0)

LSb (0)	1	2	3	4	5	6	MSb (7)
capt	tran	testmode	unused	unused	unused	unused	unused

This register is for test purposes only and should not be written during regular operation.

capt: (0) If this bit is high, the overhead capture block will be forced to capture whatever is on the STS data bus on every clock.

tran: (0) If this bit is high, the overhead capture block will be forced to transfer whatever it has captured into its internal space into the user-readable captured data space (addresses 18 - 126).

Together these two signals can be used to quickly check that all bits of the data capture space can be set high and low.

testmode: (0) If this bit is high, the chip will be put into its test mode. In this mode any frame pulse received word-aligned on the FPI(N) lines will pass through the IC as a valid frame pulse. Since all the internal blocks synchronize to the FP signal, this allows for fast testing of the IC. Normally the framing block filters out all improperly timed frame pulses. In the test mode, the LOF pin displays a characteristic waveform if the LOF circuitry is functioning properly.

Appendix 3: Probability of False Framing

The SDH standard specifies that "the algorithm used to recover from OOF shall be such that the probability for false frame recovery with a random unframed signal is no more than 10^{-5} per 250 μs time interval [8']." For the following discussion, consider the SONET stream as a long, known sequence of random bits, and choose a starting point to call bit 0. Assign negative indices to bits that precede the starting point in the order of transmission, and positive indices to those that follow it. Define a sample of length N as a set of N consecutive bits from the sequence, and shifting such that shifting the sample by M moves bit K + M into the place of bit K in the sample. Using these conventions, the state machine of the SORCC frame control block and the frame detection system scanning for the 24-bit A1A2A2 pattern implement the following algorithm to recover from OOF:

1. Compare a 24-bit sample to the framing pattern. On a match, go to (2). Otherwise, shift sample by one bit and repeat (1).
2. Shift the sample point by 311040 bits (one frame) and compare to the framing pattern. If it matches, declare in-frame. Otherwise, shift the sample by 48 and return to (1).

If this process starts in stage 1 on bit 0, the algorithm has only one chance of going in-frame in the 250 μs window. This chance corresponds to stage 1 finding a match in the first 311040 bits (1 frame) of the 622080 bits in the window, and stage 2 matching the one 24-bit sample it checks. If stage 1 does not find a match in the first frame, or if the stage 2 check does not match, there is no way to complete the framing operation in the remaining bits of the window. For future use, define the probability of going in frame in a random 622080 bit sequence starting from stage 1 at bit 0, i.e. of passing stage 1 in 311040 tests and passing stage 2 in one test, as P_{f1}.

For meeting the standard's requirement, specified "per 250μs interval," always starting the interval in stage 1 is not proper. The first 125 μs of the interval could be either

of the 125 μs "slices" from the previous discussion: a stage 2 check, or a space in which stage 1 could cause a hit. If the whole process of going in frame is treated as an atomic check with probability P1 of success, there are two cases that could cause an in-frame in this arbitrary 250 μs window: the first check could hit and block the possibility of a second check, or the first check could miss and the second could hit. The probability for this can be extracted from the "ands" and "ors": $P_{\text{false frame}} = P_{f1} + ((1 - P_{f1}) * P_{f1})$.

The problem remaining is the computation of P_{f1}. The event described by P_{f1} consists of a hit in stage 1 and a hit in stage 2, so, defining P_1 and P_2 as the probabilities of those events, $P_{f1} = P_1 * P_2$. P_2 is easy to compute; it is just the probability of a random set of 24-bits matching a fixed pattern: 2^{-24}. P_1 is equivalent to one minus the probability of not finding a single match to a 24-bit sequence in 311040 trials. The complication in calculating this value arises from the fact that, because 23 bits of each new sample are just a shifted version of values from the last, the trials are not independent. This fact can be verified by noting that, if one test produces an exact match to the framing sequence, the next one can not. Thus P_1 is not simply $(1 - (1 - 2^{-24})^{311040})$, but is best computable by developing a probability generating function from relationships between the set of all sequences not containing the 24-bit pattern and the set of all sequences ending in that pattern (but not containing it elsewhere) [17]. Extracting a particular value from this function involves expanding a power series to the term corresponding to the desired sequence length. For P1 this would be the 311040^{th} term in the series.

One can avoid the difficulty of computing that value by establishing an upper bound on P_1. The most conservative such bound, of course, is $P_1 < 1$. Using this bound, we have $P_{f1} < P_2$, and $P_{\text{false frame}} < P_2 + ((1 - P_2) * P_2) = 2^{-24} + ((1 - 2^{-24}) * 2^{-24}) = 1.19 * 10^{-7}$. A better bound comes from the fact, derived in [17], that the actual probability of finding a pattern at the end of a random sequence is highest for the case in which all trials are independent; that is, the case in which the matching pattern does not overlap itself at all.

This establishes a much more restrictive upper bound on P_1 that is still easy to compute. Using this bound:

$P_1 < (1 - (1 - 2^{-24})^{311040}) = 0.0184$

$P_2 = 2^{-24} = 5.96 * 10^{-8}$

$P_{f1} = P_1 * P_2 < 0.0184 * 5.96 * 10^{-8} = 1.09 * 10^{-9}$

$P_{false\ frame} = P_{f1} + ((1 - P_{f1}) * P_{f1}) < 1.09 * 10^{-9} + ((1 - 1.09 * 10^{-9}) * 1.09 * 10^{-9})$

$P_{false\ frame} < 2.18 * 10^{-9}$

Either of these bounds for $P_{false\ frame}$ confirms that it surpasses the 10^{-5} limit set by the standard, so the framing algorithm used in the SORCC meets the specification.

References

Many notes have two versions (1, 1') to reference both standards. The SONET reference is listed first and the SDH second where both exist.

[1] Digital Hierarchy - Optical Interface Rates and Formats Specifications (SONET), ANSI T1.105-1991, 1991.

[2] Synchronous Optical Network (SONET) Transport Systems: Common Generic Criteria, Bellcore Technical Reference TR-NWT-000253, 1991.

[3] Network Node Interface For The Synchronous Digital Hierarchy, CCITT Recommendation G-708, 1988.

[4] Frame Structure: ANSI T1.105-1991, p. 43.
Overhead diagram: ANSI T1.105-1991, p. 59.
Overhead byte description: ANSI T1.105-1991, p. 13.

[5] "Extract From The Report Of Working Party XVIII, " ISDN Experts Of Study Group 18, SWP-3 Recommendations on CCITT G-708, 1993, pp. 14 - 19.

[6] Bellcore TR-NWT-000253, ch. 4.

[7] ANSI T1.105-1991, p. 11.

[8] Bellcore TR-NWT-000253, p. 5-29.

[8'] Characteristics Of Synchronous Digital Hierarchy (SDH) Multiplexing Equipment Functional Blocks, CCITT Recommendation G-783, 1990, p. 11.

[9] ANSI T1.105-1991, pp. 22 - 23, 72.

[9'] Synchronous Multiplexing Structure, CCITT Recommendation G-709, 1988, p. 130.

[10] The structure of an OC-48 signal in terms of lower rate signals is shown in Figure 5. It is defined in the following references:
ANSI T1.105-1991, p. 22.
CCITT G-709, pp. 128 - 129.

[11] Bellcore TR-NWT-000253, p. 6-37.

[11'] CCITT G-783, p. 14.

[12] ANSI T1.105-1991, p. 12 - 13.

[12'] CCITT G-709, pp. 119 - 120.

[13] ANSI T1.105-1991, p. 29.

[13'] CCITT G-783, p. 14.

[14] Bellcore TR-NWT-000253, p. 6-30.

[14'] CCITT G-783, p. 11.

[15] Bellcore TR-NWT-000253, p. 6-29.

[16] DooWhan Choi, "Parallel Scrambling Techniques For Digital Multiplexers," AT&T Technical Journal, September / October 1986, Vol. 66, pp. 123 - 125.

[17] Ronald Graham, Donald Knuth, and Oren Patashnik, A Foundation for Computer Science: Concrete Mathematics (New York: Addison-Wesley) pp. 389 - 396.

Bibliography

Characteristics of Synchronous Digital Hierarchy (SDH) Multiplexing Equipment Functional Blocks. CCITT Recommendation G-783. Geneva: The International Telegraph and Telephone Committee, 1990.

Choi, DooWhan. "Parallel Scrambling Techniques For Digital Multiplexers." AT&T Technical Journal September / October 1986, Vol. 66: 123 - 125.

Digital Hierarchy - Optical Interface Rates and Formats Specifications (SONET). ANSI T1.105-1991. New York: American National Standards Institute, 1991.

Graham, Ronald, et al. A Foundation for Computer Science: Concrete Mathematics. New York: Addison-Wesley, 389 - 396.

Layer 1 In-Service Digital Transmission Performance Monitoring. ANSI T1M1.3/93-005R1. New York: American National Standards Institute, 1993.

Network Node Interface For The Synchronous Digital Hierarchy. CCITT Recommendation G-708. Melbourne: The International Telegraph and Telephone Committee, 1988.

Synchronous Multiplexing Structure. CCITT Recommendation G-709. Melbourne: The International Telegraph and Telephone Committee, 1988.

Synchronous Optical Network (SONET) Transport Systems: Common Generic Criteria. TR-NWT-00253. New Jersey: Bellcore, 1991.